Konzeptverifizierung und Implementation eines hardwareunabhängigen, zentralen Konfigurations- und Management Systems von Workstations in einem industriellen Umfeld

Martin Hald

Konstanz, 15.03.2004

DIPLOMARBEIT

Zusammenfassung (Abstract)

Thema:	**„Konzeptverifizierung und Implementation eines hardwareunabhängigen, zentralen Konfigurations- und Management Systems von Workstations in einem industriellen Umfeld“**
Diplomand:	Martin Hald
Firma:	ABB Schweiz AG Utility Automation Bruggerstr.72 CH-5400 Baden
Betreuer:	Professor Dr.-Ing. Jürgen Neuschwander Diplom Ingenieur (Uni) Michael Layes
Abgabedatum:	15.03.2004
Schlagworte:	OnTechnology , CCM, OnCommand, automatische, Deployment Werkzeuge, Softwareverteilung, unattended Setup, RIS, Sysprep, Autoyast, Ignite UX

Zur Verringerung der Kosten und des Aufwands zur Installation, Update und Wartung von Workstations in einem industriellen Umfeld, sollte ein System zur zentralen, hardwareunabhängigen und vollautomatischen Installation dieser Workstations eingeführt werden.
Hierzu mussten zuerst einmal die Möglichkeiten für eine vollautomatische Installation der verschiedenen Betriebssysteme und Anwendungen betrachtet werden. Nach dem Aufbau der Hardware folgte daraufhin die Installation des Deployment Systems der Firma OnTechnology (OnCommand). Im Anschluss daran wurden die Skripte für die Betriebssystem- und Anwendungsinstallation erstellt. Diese Skripte wurden anschließend zur besseren Handhabung zu Profilen zusammengefasst. Im nächsten Schritt wurden bestehende Clients in das System integriert. Daraufhin wurde das Sicherheitskonzept des Systems betrachtet, mit dem die Möglichkeit besteht bestimmten Benutzern nur bestimmte Rechte zu vergeben.

In weiteren Teilen der Diplomarbeit werden die bisherigen Probleme bei der Firma und das erhoffte Ziel, die Möglichkeiten des Systems, sowie die Probleme bei der Umsetzung beziehungsweise des Systems, dargestellt und im Fazit zusammengefasst.

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Informatiker (FH)

an der

Fachhochschule Konstanz

Hochschule für Technik, Wirtschaft und Gestaltung

Fachbereich Informatik

Studiengang TI

Thema: **Konzeptverifizierung und Implementation eines hardwareunabhängigen, zentralen Konfigurations- und Management Systems von Workstations in einem industriellen Umfeld**

Diplomand: Martin Hald, Küssnacherstr.27, 79801 Hohentengen-Lienheim

1. Prüfer: Professor Dr.-Ing. Jürgen Neuschwander

2. Prüfer: Diplom Ingenieur (Uni) Michael Layes
ABB Schweiz AG
Utility Automation
Bruggerstr.72
CH-5400 Baden

Abgabedatum: 15.03.2004

Ehrenwörtliche Erklärung

Hiermit erkläre ich, Martin Hald, geboren am 13.12.1975 in Konstanz, ehrenwörtlich,

(1) dass ich meine Diplomarbeit mit dem Titel:
„Konzeptverifizierung und Implementation eines hardwareunabhängigen, zentralen Konfigurations- und Management Systems von Workstations in einem industriellen Umfeld“
bei der ABB Schweiz AG Power Automation unter Anleitung von Professor Dr.-Ing. Jürgen Neuschwander selbständig und ohne fremde Hilfe angefertigt habe und keine anderen als die angeführten Hilfen benutzt habe;

(2) dass ich die Übernahme wörtlicher Zitate, von Tabellen, Zeichnungen, Bildern und Programmen aus der Literatur oder anderen Quellen (Internet) sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Konstanz, 15.03.2004 _____

Unterschrift

1. Allgemeine Beschreibung 3
1.1 Einleitung 3
1.2 Gegenwärtige Situation ABB Schweiz AG 4
1.3 Probleme ABB Schweiz AG 4
1.4 Ziel 5

2. Lösungsansätze 7
2.1 Grundvorrausetzungen 7
2.2 Allgemein 8
2.3 Betriebsysteme 10
2.3.1 Microsoft Windows 11
2.3.1.1 Windows 2000/XP 11
2.3.1.2 Windows NT 19
2.3.2 Suse Linux 20
2.3.3 HP Tru64 Unix 25
2.3 Anwendungen 27
2.4 Systeme zur automatischen Softwareverteilung 30
2.5 Verwendete System 32
2.5.1 Über OnTechnology 33
2.5.2 Produkte 33
2.5.3 Technology des Systems 34
2.5.4 Vorteile des Systems 36

3. Umsetzung 39
3.1 Systemaufbau 39
3.1.1 Server 39
3.1.2 Raid 40
3.1.3 Server Betriebssystem 51
3.1.3 CCM System Aufbau 54
3.2 Skriptierung 59
3.2.1 Möglichkeiten der Skripterstellung für Anwendungen (CCM) 59
3.2.1.1 MSI Package Wizard 60
3.2.1.1.1 Verwendung 61
3.2.1.1.2 Installation 61
3.2.1.1.3 Paketerzeugung 61
3.2.1.1.5 Verbesserungen in Config.swp 70
3.2.1.1.6 Testen 70
3.2.1.1.7 Probleme 71
3.2.1.2 Development Studio 71
3.2.1.2.1 Verwendung 71
3.2.1.2.2 Installation 71
3.2.1.2.3 Paketerzeugung 72
3.2.1.2.4 Testen 79
3.2.2 Config.swp 81
3.2.3 Package Import Tool 83
3.3 Umsetzung 84
3.3.1 Betriebsystempakete (Windows NT/2000/XP) 84
3.3.2 Anwendungspakete 88
3.4 Probleme bei der Umsetzung 88
3.4.1 Standardprobleme 88
3.4.2 Spezielle Probleme bei Paketen 92

4. Sonstige Tätigkeiten 100
4.1 Integration bestehender Clients 100
4.1.1 Autodiscover (OnWards) 100
4.1.2 Probleme Autodiscover 101
4.2 Zusammenfassung von Konfiguration 102
4.3 Zugriffsberechtigungen 103
4.3 Hinweismeldung bei Fehler 104

5.Fazit 106

Anhang A – Eigene Pakete 108
Anhang B – Quellcode position.vbs 113
Anhang C – Quellcode runas.vbs 117
Anhang D – Quellcode startrun.exe 118
Anhang E – Quellcode runonce.bat 119
Anhang F – Testbericht PC-Professional 06/2003 121
Anhang G – Treiberintegration 123

Literaturverzeichnis 125

Abbildungs- und Tabellenverzeichnis 126

Abkürzungsverzeichnis 128

1. Allgemeine Beschreibung

1.1 Einleitung

Eine der kostenintensivsten Aufgaben die von IT Personal durchgeführt wird, ist die Verteilung, beziehungsweise Installationen von Betriebssystemen auf bestehende oder neue Clientcomputer. Momentan verlieren viele Unternehmen Zeit und Geld für die Installation der neuesten Versionen der Betriebssysteme, Updates und Applikationen.

Wie entstehen diese Kosten?

Oft wird dies manuell von einem Computerspezialisten durchgeführt, welcher dabei direkt beim Rechner vor Ort ist. Bei dieser Vorgehensweise entstehen dem Unternehmen Kosten zwischen 180 und $ 300 für die Installation eines Rechners. Der Gesamtaufwand pro Jahr für einen Windows- PC beträgt im Durchschnitt 6000,- Euro [17].

Ein weiterer Grund ist die Installation einer großen Anzahl von gleichartigen Systemen. Hat man eine große Anzahl von Client- oder Server-Systemen, z.B. in einem hochgeschwindigkeits- Cluster, welche auf gleiche Art installiert werden müssen, so wäre es zu aufwendig jeden dieser einzelnen Rechner von hand zu installieren.

Das Aufsetzen eines Server-Systems, die Auswahl der Software und die Konfiguration des Netzwerks ist eine anspruchsvolle Aufgabe, die normalerweise von ausgebildeten System-Administratoren ausgeführt wird. Auf der anderen Seite aber haben diese hochqualifizierten und gut bezahlten Menschen, nicht genügend Zeit und die Installationen muss oft an verschiedenen Orten durchgeführt werden.

Ein weiterer Grund für die hohen Kosten kann sein, dass mit immer neueren Betriebssystemen, die Systemplattformen alle 2-3 Jahre gewechselt werden.

Weiterhin ist es möglich, dass eine Installation an einem weit entfernten Ort, an dem es kein geschultes Personal gibt oder die Reisekosten zu hoch wären, durchgeführt werden muss.

Aus den oben genannten Gründen, wird sich immer mehr mit dem Thema der automatischen Softwareverteilung auseinandergesetzt.

1.2 Gegenwärtige Situation ABB Schweiz AG

ABB Utility Automation Systems PTUP liefert Systeme zur Kraftwerksautomatisierung, und zur Beurteilung, Auswertung, Simulation und Optimierung des thermischen Prozesses. Die Arbeitsbereiche erstrecken sich über thermische Kraftwerke, Wasserkraftwerke, Wasserversorgung und Kehrichtverbrennung.

Für die Regelung, Steuerung und Schutz wird eine ABB eigene Geräte Plattform implementiert. Diese Plattform enthält Controller, Ein- und Ausgabegeräte, sowie Kommunikationsbusse und –geräte.

Die Bedienung der Anlagen erfolgt im weiteren Sinne über handelsübliche Workstations. Hierbei kommen sowohl Unix-, wie auch PC basierende Workstations auf Windows oder Linux Basis zum Einsatz.

1.3 Probleme ABB Schweiz AG

Für die oben erwähnten Workstations ist die Betriebssystem- und Applikationssoftware in grösserer Anzahl zu installieren. Diese Workstations sind über ihre Lebensdauer zu pflegen, sowie im Falle eines Systemcrashes wieder herzustellen. Das gleiche gilt für die beim Kunden im Kraftwerk installierten Arbeitsstationen.

Im Moment wird dies durch das Erstellen von Imagefiles bzw. binären Abbildern einer Harddisk, mittels Symantec Norton Ghost gehandhabt. Dies hat jedoch einige gravierende Nachteile.

- Die Setups der Workstations müssen hardwarespezifisch aufgesetzt werden
- Man erhält eine Vielzahl von CDs mit den verschiedenen Images, wodurch ein enormer Verwaltungsaufwand entsteht
- Mit längerer Lebenszeit erhöht sich die Komplexität und Vielfalt der Abbilder
- Es entsteht ein hoher Wartungsaufwand, der sich stets erhöht
- Treten Probleme bei PCs auf, die mittels Ghost Abbildern aufgesetzt wurden, fordern viele Softwarehersteller als 1.Massnahme oft ein Originalsetup, bevor weiter Hilfestellung erfolgt
- Diverse Hardware muss für das Erstellen und die Pflege der Abbilder vorhanden sein

Aus den oben genannten Gründen erwachsen in absehbarer Zeit grössere Zeitprobleme bei der Installation, sowie bei der Wartung der Workstations

1.4 Ziel

Das Ziel besteht darin, den Aufwand zur Installation und Wartung der Workstations zu verringern. Dabei soll es möglich sein zentral, hardwareunabhängig und mit möglichst wenig Aufwand, die Workstations zu installieren und warten.
Zu diesem Zweck wurde ein System evaluiert, dass diese Anforderungen erfüllen kann. Mit diesem System sollen eine Reihe der oben genannten Probleme eliminiert werden. Die wesentlichen Merkmale sind dabei:

- Die Installation der Systeme erfolgt auf Basis des Originalsetups der Software
- Es gibt eine (oder mehrere) Datenbasen die auf einem (oder mehreren) Server liegen
- Konfigurierte Setups laufen komplett und vollautomatisch mit allen Applikationen (so genanntes „unattended Setup“) ab

- Remote Setups können vom ABB Server via Remote Access ausgeführt werden
- Erfolgreiche Installation kann mit Hilfe der Logfiles auf dem Server verifiziert und dokumentiert werden (=> gleichzeitiges Qualitätsdokument)
- Auf dem Server kann ein Inventar der installierten bzw. eingebundenen Computer gepflegt werden
- Möglichkeit der automatischen Inventarisierung durch Fernabfragen via LAN, WAN, RAS und Internet
- Identifizierung der Hard- sowie Software. Vollständige Erfassung aller Netzwerkgeräte, inklusive deren Hard- und Softwareausstattung
- Im Gegensatz zu Ghost Abbildern kann jeder ausgelieferte Computer mit einer eigenen Lizenz, in Übereinstimmung mit den Lizenzpapieren ausgeliefert werden

2. Lösungsansätze

Im folgenden Kapitel wird auf allgemeine, betriebssystemspezifische und softwaretechnische Möglichkeiten zur automatischen Softwareverteilung eingegangen.

2.1 Grundvorrausetzungen

Ist eine vollautomatische Verteilung des Betriebssystem gewünscht, so muss der Client Rechner aus der Ferne aufgeweckt und auch ohne installiertes Betriebsystem gebootet werden können.

Damit der Rechner aus der Ferne aufgeweckt werden kann, muss dieser das so genannte „Wake up on Lan (WoL)“ unterstützen. Dabei wartet die Netzwerkkarte auf ein bestimmtes Paket dass an sie adressiert ist. Erkennt die Karte einen solchen Frame, so wird der PC Eingeschalten. Hiervon gibt es zwei Spezifikationen. Zum einen die AMD-Spezifikation „Magic Paket“ und zum anderen die IBM „Wake-Up Frame“. Bei AMD handelt es sich um eine bestimmte Byte-Folge, die sich aus einer Synchronisationsfolge von sechs Bytes (FF), gefolgt von der MAC-Adresse (wird 16 mal wiederholt) des Zielrechners zusammensetzt. Die IBM Spezifikation hingegen hat eine andere Vorgehensweise. Hier werden die Bytes eines Frames anhand einer Maske untersucht, und anschliessend mit einer Liste von Beispiel-Frames verglichen.

Um einen Rechner der noch kein Betriebsystem besitzt eine gestartet werden kann, wurde von der Industrie der Pre-Boot Execution (PXE) Standard eingeführt. Mit diesem Standard, hat man die Möglichkeit einen Client Rechner über das Netz zu booten, sowie diverse Netzwerkparameter wie zum Beispiel PXE- Informationen, IP-Adresse, Serveradresse mitzuteilen. Anfragen an den Server werden immer vom Client initiiert. Mittels dieser Technologie besteht die Möglichkeit jedes System, was sich mit einer Bootdiskette starten lässt, über das Netzwerk zu starten. Dazu muss lediglich aus einer Bootdiskette ein Boot-Image erzeugt werden.

Eine weitere Möglichkeit für das Booten via Netzwerk, ist der Einsatz von Bootp-Images (Abkürzung für Bootstrap Protocol), welche eine statische Zuordnung von Konfigurationsparametern (TCP/IP) ermöglichen. Diese werden mit Hilfe von Trivial File Transfer Protocol Daemon (TFTP) an den Client übertragen.

Auch die Anforderungen an die Netzwerkinfrastruktur sind stark von den eingesetzten Softwareverteilungswerkzeugen abhängig. Entscheidende Faktoren sind zum einen die Bandbreite, sowie die Netzwerkstruktur (geroutetes oder geswitchtes Netzwerk).
Arbeitet die verwendete Software mit Broadcast (alle Geräte im Subnetz reagieren auf diese Adresse), so sollte dieses nur in gerouteten Netzwerkumgebungen eingesetzt werden, da die Pakete an alle Geräte gesendet werden.
Unterstützt die Software Multicast (Informationen werden nur an Empfänger geschickt, die diese auch erhalten möchten), so müssen die Empfänger einer so genannten Multicastgruppe angehören und die Auswertung dieser Informationen auch von den Routern unterstützt werden. Da IP-Multicast auf UDP (User Datagram Protocol) beschränkt ist, und somit die Übertragung verbindungslos arbeitet, besteht die Möglichkeit, dass Pakete verloren gehen, in falscher Reihenfolge ankommen oder ihr Ziel gar nicht erreichen.
Verwendet die Software Unicast (Information wird an einen Empfänger gesendet), so ist ein geswitchtes Netzwerk von Vorteil. Hierbei werden die Informationen nur an den Empfänger gesendet der diese auch wirklich benötigt. Ein Vorteil ist vielleicht die Möglichkeit den Verteilserver mit einer schnelleren Netzwerkkarte bzw. Netzwerkverbindung auszustatten, damit dieser mehrere Clients gleichzeitig mit höherer Geschwindigkeit versorgen kann.
Quellen : [3] [4] [14]

2.2 Allgemein

Grundsätzlich existieren drei Verfahren Software auf einem Client Software zu installieren. Die erste Methode benutzt die originale Setuproutine des Herstellers, welches dann ferngesteuert automatisch ausgeführt wird. Eine weitere Methode ist

das so genannte Snapshot Verfahren. Bei diesem werden die Differenzen vor und nach einer Installation aufgezeichnet und diese Unterschiede dann auf dem Client nachgeführt. Die letzte Möglichkeit besteht darin, mittels Imaging- Verfahren (auch Cloning genannt) zu arbeiten. Bei dieser Methode wird der komplette Inhalt einer Festplatte auf einen anderen Rechner kopiert.

Vor- und Nachteile der Verfahren:

	Ferngesteuertes Original Setup	**Snapshot Verfahren**	**Imaging- / Cloning Verfahren**
Vorteile	• Programm wird so installiert wie es der Hersteller erwartet • Keine Probleme wenn Systemkomponenten aktualisiert werden müssen • Hardwareunabhängig	• Einfache Aufzeichnung der Installation und Konfiguration • Schnelle Erstellung der Skripte möglich • Hardwareunabhängig	• Schnellstes Verfahren, da Daten nur kopiert werden müssen und nicht von Grund auf installiert werden
Nachteile	• Initialaufwand sehr aufwendig da alle möglichen Optionen abgefangen werden müssen • Dadurch entstehen auch hohe Kosten • Installation dauert im Vergleich mit Imaging sehr lange	• Eventuelle Probleme bei bereits installierten Dateien -> alte oder neue Anwendung funktionieren nicht korrekt • Sollten ähnliche Softwarekonfigurationen auf den Rechnern vorhanden sein • Aufwand für das Testen der Skripte kann sehr hoch werden • Installation dauert im Vergleich mit Imaging sehr lange	• Hardware abhängig • Muss identische Rechner haben, ansonsten muss für jeden Rechnertyp ein Image erstellt werden • Grosser Verwaltungsaufwand und Speicherplatzbedarf für die Images

Tabelle 1 - Vor/Nachteile Installationsmethoden

Weiterhin wäre es möglich einen Interpreter (z.B. Pearl) auf dem noch fast leeren System (ohne Windows Betriebssystem aber z.B. mit MS-DOS) zu installieren. Mit dessen Hilfe könnten diverse Aufgaben wie z.B. Änderungen von Dateien/Konfigurationen, Erweiterungen oder andere Aufgaben zur Laufzeit erledigt werden [12].

2.3 Betriebsysteme

Allgemein kann der hardwareunabhängige Autoinstallationsprozess bzw. der normale Installationsprozess eines Betriebssystems in drei Phasen eingeteilt werden. Diese drei Phasen sind Vorbereitung, Installation und Konfiguration. Die folgende Abbildung zeigt wie diese Phasen aufeinander folgen und was in diesen getan wird:

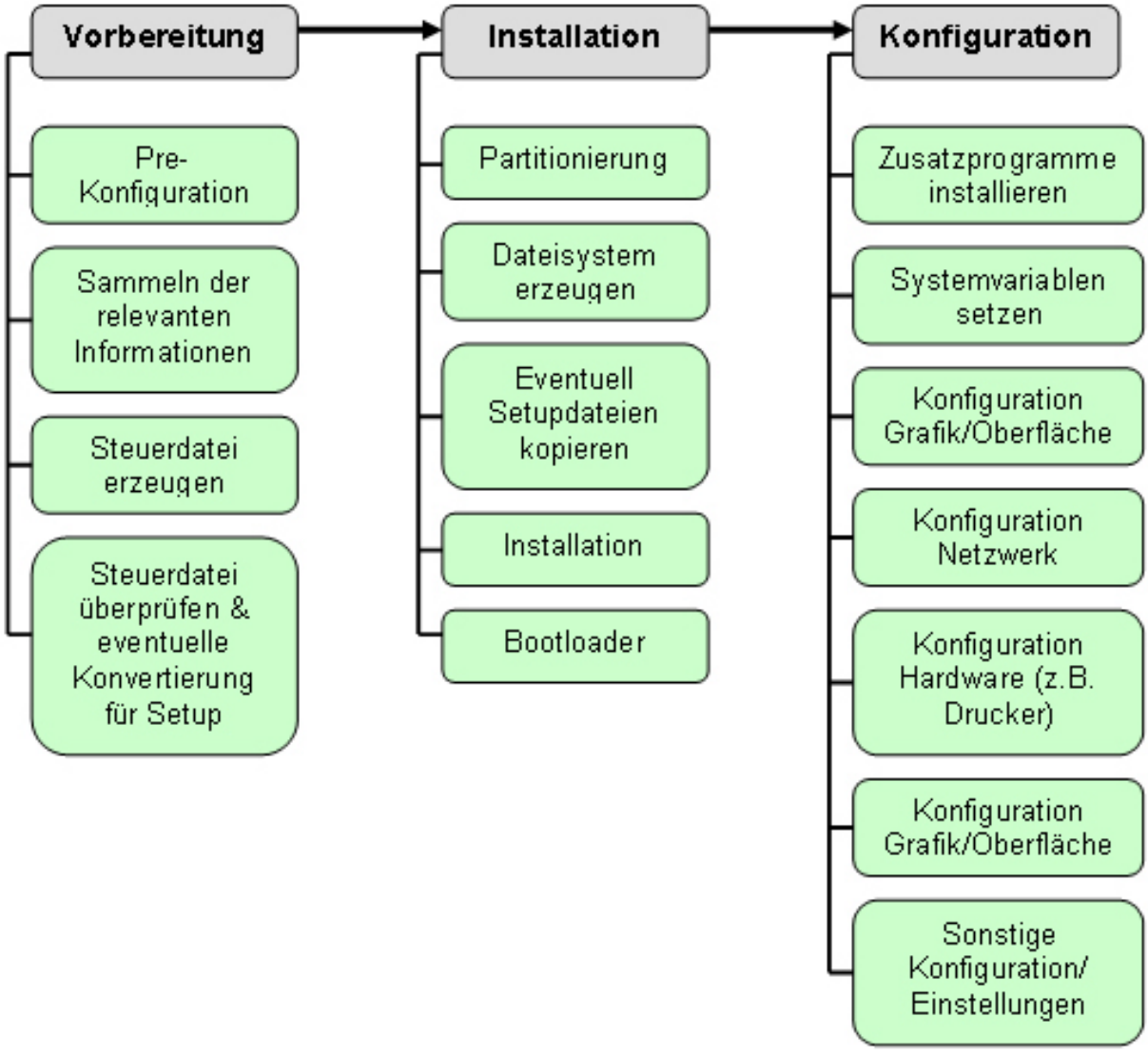


Abbildung 1 - Phasen während der Betriebsysteminstallation

In der Vorbereitungsphase werden alle relevanten Informationen des Zielssystems gesammelt und in entsprechende Anweisungen für die Steuerdateien konvertiert.

Diese Steuerdatei wird auf das Zielsystem übertragen, auf welchem dann aus dieser die Einstellungen ausgelesen werden.
Während der Installation Phase werden diese Einstellung ausgewertet und das System anhand dieser installiert.
In der 3. Phase, der Konfiguration, werden dann die systemspezifischen Einstellungen wie z.B. Netzwerkeinstellungen getätigt und eventuelle Zusatzsoftware installiert.

2.3.1 Microsoft Windows

In den folgenden Unterpunkten wird auf die Möglichkeiten zur vollautomatischen/unbeaufsichtigten Installation der Windows Betriebssysteme eingegangen. Da sich die Installationsmethoden von Windows 2000 und Windows XP nur in Details (andere Parameternamen) unterscheiden, werden diese beiden Systeme zusammengefasst betrachtet.

2.3.1.1 Windows 2000/XP

Für eine unbeaufsichtigt Installation von Windows 2000/XP gibt es mehrere Verfahren. Die möglichen Verfahren sind unattended Setup, SYSPREP und RIS. Diese 3 Möglichkeiten der unbeaufsichtigten Installation werden im Folgenden beschrieben.

SYSPREP

Bei diesem Verfahren werden Image Files (Abbilder) erzeugt. Dabei muss zuerst ein so genanntes Master Image erzeugt werden. Dies geschieht auf folgende Weise:

- Festplatte komplett löschen
- FAT 32 Partition mit der am geringsten vorkommenden Plattengröße erzeugen
- Partition formatieren
- Netzwerk Verbindung herstellen

- Unattended Windows 2000 professionell Setup Ausführung (<Setuppfad>:\SYSTEMSW\W2P_DE\W2PINST.BAT)
- Zusätzlich gewünschte Applikationen installierten

Durch Ausführungen der SYSPREP Batchdatei, wird das vorbereitete SYSPREP Tool auf die Festplatte kopiert

- SYSPREP ausführen
- Computer neu gestartet werden

Zum Abschluß wird ein Image von der Festplatte erzeugt. Dies kann z.B. mit PowerQuest DriveImage oder Norton Ghost erfolgen.

Nachdem man nun das Master Image auf diese Weise erzeugt wurde, kann nun der Client mittels Bootdiskette via Netzwerkverbindung aufgesetzt werden. Hierzu sind folgende Schritte zu durchlaufen:

PC mit Bootdiskette starten. Dabei werden Treiber für die Netzwerkverbindungen geladen und die Verbindung hergestellt.
Anschliessend wird das Image restauriert. Dies geschieht mittels einer Batchdatei (restore.bat). Dieser Datei muss der Imagename als Parameter mitgegeben werden
Nun hat man die Wahl ob die Benutzerdaten sofort eingetragen werden sollen oder diese erst vom Benutzer eingetragen sollen. Will man die Daten sofort eintragen, so muss der Computer neu gestartet werden. Andernfalls wird er in diesem Zustand ausgeliefert, wie es bei einem Neukauf eines Computers oft der Fall ist.
Hat man sich für die sofortige Konfiguration entschlossen, so erscheint eine Mini - Benutzeroberfläche, in welcher die Daten wie Benutzername, Computername, Kennwort und Arbeitsgruppe/Domäne eingegeben werden müssen.
Jetzt muss der Computer neu gestartet werden. Nach dem Neustart erfolgt eine automatische lokale Anmeldung ohne Netzwerkautorisierung. Die speziellen Netzwerkeinstellungen müssen anschliessend noch manuell eingetragen werden.

Somit ist die Softwareverteilung mit SYSPREP abgeschlossen. Diese Verteilung entspricht einer Festplattenduplizierung, die zur Masseninstallation von

hardwaregleichen Computern eingesetzt werden kann. Gleichzeitig ist dieses Verfahren von der Geschwindigkeit her gesehen das Beste. Weiterhin wird auch kein spezieller Server benötigt. Es spielt keine Rolle ob die Images auf einem Windows, UNIX oder sonstigem System abgelegt werden. Einzige Vorrausetzung ist dass diese im Netzwerk freigegeben sind.

Ein großer Nachteil liegt jedoch in der Hardwareabhängigkeit dieses Verfahrens. So wird beim Wechsel einer Hardwarekomponente ein neues Master Image nötig. Somit erhält man mit im laufe der Zeit eine Vielzahl von Image Dateien, für welche ein erheblicher Verwaltungsaufwand betrieben werden muss, sowie eine Unmenge an Speicherplatz benötigt wird.
Ein neues Image wird auf jeden Fall bei folgenden Wechseln nötig:

- Architektur – ACPI- System oder nicht
- Single – Mutiprozessorsystem
- Chipsatzwechsel

Des Weiteren werden bei diesem Verfahren verschiedene Einstellungen wie z.B. Computer Name oder gar Security Identifier (SID) mit in das Image aufgenommen. Diese müssen nach der Installation manuell geändert werden.

Remote Installation (RIS)

Remote Installation (im folgenden Text mit RIS abgekürzt) ermöglicht die Installation der Client-Rechner über das Netzwerk. Hierzu wird ein Abbild (Image), welches das Betriebssystem sowie alle installierten Applikationen enthält, auf dem RIS-Server abgelegt. Dieses Image kann dann mittels netzwerkfähiger Boot Diskette oder mittels eines PXE-fähigen Boot-Proms über das Netzwerk durch das entfernte booten verteilt werden. Hierbei können die Rechner automatisch konfiguriert werden. So kann zum Beispiel der Rechnername für jeden Rechner individuell eingestellt werden.
Im Gegensatz zu unter Punkt 2.3.1.1 beschriebenen SYSPREP, arbeitet RIS weitestgehend Hardwareunabhängig und lässt sich auch individuell konfigurieren.

Bei RIS wird ein neutrales Image erstellt. Beim zurückspielen des Image werden die notwendigen Treiber installiert, eine neue SID vergeben und bei entsprechender Konfiguration z.B. Computername, IP usw. individuell vergeben. Die einzige Voraussetzung bzgl. der Hardware ist, ein identischer Hardware Abstraction Layer (HAL). Dies bedeutet z.B. dass ein auf einem IDE- System erstelltes Image nicht auf einem SCSI- System installiert werden kann.
Um mit diesem Verfahren zu arbeiten, muss der RIS-Service zunächst auf den Server installiert werden. Gleichzeitig ist eine Windows 2000 CD von Nöten, da bei diesem Schritt auch gleich das Grundimage hinterlegt wird.

Der Ablauf einer RIS Installation in einer DHCP Umgebung soll anhand des folgendem Schaubild erläutert werden:

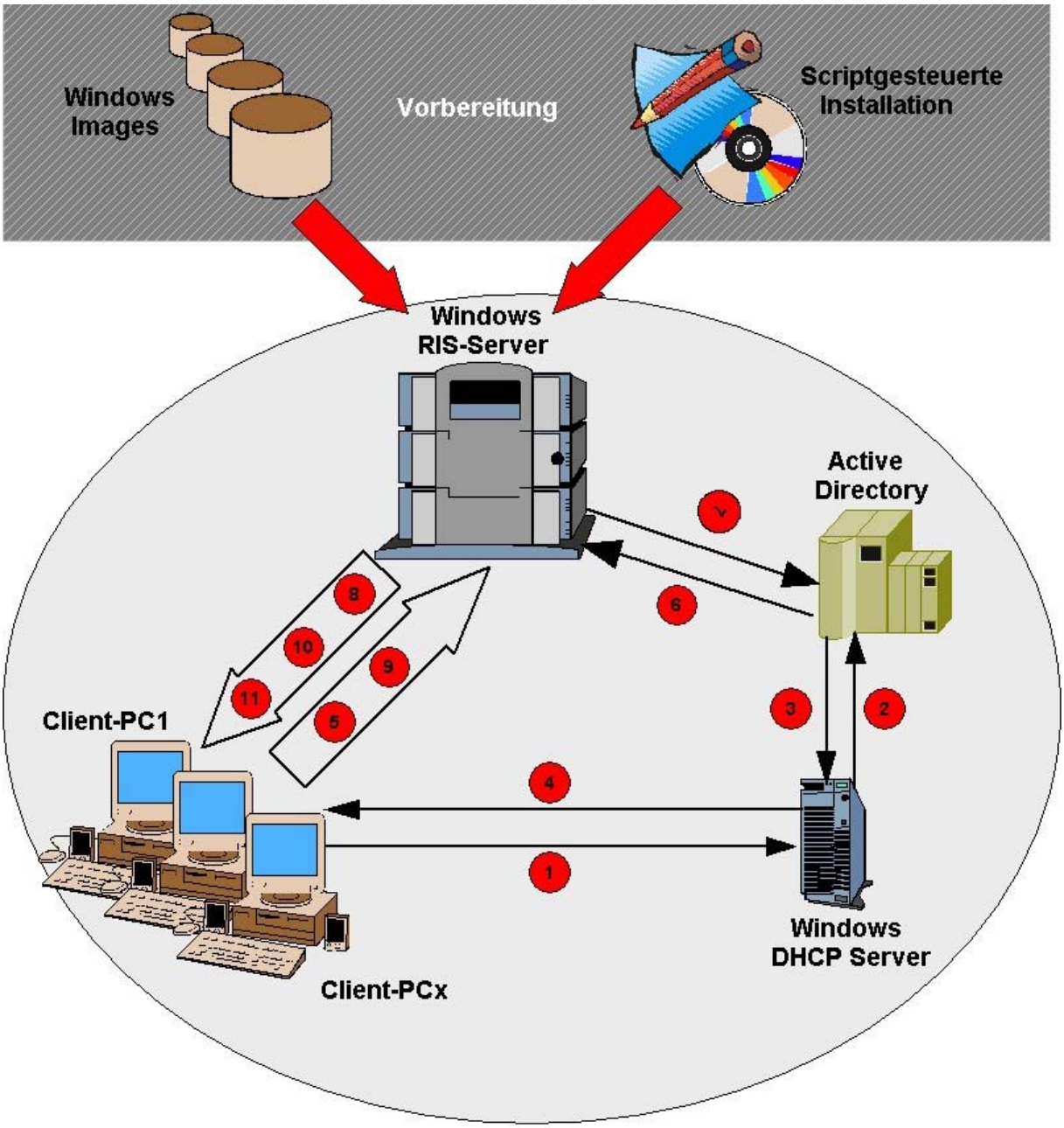


Abbildung 2 - Ablauf einer RIS Installation

Erklärung der Schritte (1-11):

1. Client sendet DHCP Request an den DHCP Server
2. Der DHCP Server führt daraufhin einen so genannten RIS-Lookup im Active Directory (Datenbank, vergleichbar mit einem Telefonbuch) aus um die (Standort-)Daten des RIS-Servers zu erhalten
3. Der DHCP-Server erhält daraufhin der RIS-Standort
4. Dem Client wird eine IP zugeteilt, sowie diverse PXE Optionen

5. Daraufhin meldet sich dieser am RIS-Server an
6. Der RIS-Server überprüft daraufhin im Active Directory ob der Client im Active Directory eingetragen ist (Authentifizierung)
7. Der RIS-Server erhält daraufhin die im Active Directory eingetragenen Eigenschaften wie z.B. Computer-, Usereigenschaften (Authorisierung)
8. Daraufhin sendet er dem Client die Auswahlmöglichkeiten für die Installation (vorhandenen Betriebssystem Images)
9. Am Client wird daraufhin die Betriebsystemauswahl getroffen und dem Server mitgeteilt
10. Daraufhin erfolgt der Datentransfer des Betriebssystems
11. Im letzten Schritt wird dem Client noch das Installationsscript übertragen und die Installation beginnt

Anmerkung: DHCP- und RIS-Server können natürlich auch auf ein und dem selben Rechner installiert werden, jedoch muss ein DHCP-Server vorhanden sein.

RIS Vor- und Nachteile:

- Wenn eine Windows 2000 (oder neuere) Umgebung vorhanden ist, müssen keine weiteren Tools zur Softwareverteilung angeschafft werden.
- Die Installation wird über Antwortdateien gesteuert
- Nahezu Hardwareunabhängig
- Funktioniert erst ab Windows 2000 Server
- Windows 2000 Server muss vorhanden sein

Unattended Setup

Als letzte Möglichkeit gibt es das so genannte "unattended Setup ". Bei dieser Möglichkeit wird eine so genannte Antwortdatei erstellt, in welcher die Einstellungen (auch Antworten genannt) für das Setupprogramm gespeichert werden. Die Bezeichnung Antwort kommt daher, da bei der manuellen Installation von Windows, das Setup immer wieder anhält und ein Dialog erscheint der eine Frage stellt (z.B.

Wo soll Windows installiert werden). Daraufhin ist man gezwungen eine Antwort zu geben.
Eine solche Antwortdatei, welche als einfache Textdatei abgespeichert wird, hat generell folgendes Format:

```
[Abschnitt 1]
Schlüsselname1=Schlüsselwert1
Schlüsselname2=“Irgendein Wert“
……

[Abschnitt 2]
……
```

Der Einfachheit halber, kann gesagt werden dass eine solche Datei aus Abschnitten (in Klammern eingeschlossen) und den zugehörigen Einträgen (auch Einstellungen genannt) besteht.
Um dies besser zu verdeutlichen, dient folgender Ausschnitt aus einer solchen Antwortdatei. Er enthält die Einstellungen für die Netzwerkkonfiguration und die Benutzerdaten.

```
; der folgende Abschnitt enthält die Antworten für die Benutzerdaten
[UserData]
FullName=“Max Mustermann“
OrgName=“Muster AG“
ComputerName=“Musterrechner 1“
ProductKey=“11111-22222-33333-44444-55555“

; der folgende Abschnitt enthält die Antworten für die Netzwerkkonfiguration (TCP/IP, ; statische IP Adresse, Gateway und Subnetzmaske für die Netzwerkkarte 1)
[Networking]
[Identification]
     JoinWorkgroup = „MeineArbeitsgruppe“

; Anzahl der installierten Netzwerkkarten
[NetAdapters]
     Adapter01 = params.Adapter01

; dient zur Identifikation der Netzwerkkarte anhand der PNP-Id ; des Herstellers (Stern bedeutet erste erkannte Karte)
 [params.Adapter01]
     INFID = *
```

```
; zu installierende Netzwerkprotokolle
[NetProtocols]
     MS_TCPIP = params.MS_TCPIP

; binden der Netzwerkprotokolle an die Netzwerkkarte
[params.MS_TCPIP]
     AdapterSections =params.MS_TCPIP.Adapter01

; Einstellungen für das Netzwerkprotokoll (hier TCP/IP von Karte 1)
[params.MS_TCPIP.Adapter01]
     DHCP = no
     IPAddress = 192.168.0.10
     SpecificTo = Adapter01
     SubnetMask = 255.255.255.0
     DefaultGateway = 192.168.0.1
     WINS = no
```

Alle unterstützten Parameter in einer solchen Antwortdatei sind im Microsoft Dokument „Windows 2000 Guide to Unattended Setup“ zusammengefasst. Dieses Dokument kann auf der Microsoft Developers Homepage heruntergeladen werden.

Zum Starten der automatischen Installation, muss die so erzeugte Antwortdatei nur noch der Setuproutine mitgegeben werden bzw. dem Setup durch Parameter mitgeteilt werden dass es sich um ein unattended Setup handelt.

Aufruf:

```
c:\i386\winnt /s:c:\i386 /unattend:unattended.txt
```

Vergleich der Verfahren

Verfahren	*SYSPREP*	*RIS*	*unattended*
Geschwindigkeit	+	o	-
Unterschiedliche Hardwareunterstützung	-	+	+
Initialaufwand/Kosten	+	-	o
Benötigt Windows 2000 Server	nein	ja	nein

+ = gut o = mittel - = schlecht

Tabelle 2- Vergleich Sysprep,RIS & unattended Setup

2.3.1.2 Windows NT

Auch unter Windows NT existiert die Möglichkeit durch ein so genanntes „Unattended Setup“ einen Server oder eine Workstation vollautomatisch, d.h. ohne notwendigen Eingriff des Administrators, zu installieren. Für die Installation von Standardkomponenten wie zum Beispiel Adaptec Controller oder Netzwerkkarten ist dies ein einfaches Werkzeug. Leider werden bei dieser Art nicht alle Komponenten die es unter NT gibt unterstützt. Es besteht jedoch eine Möglichkeit, wie man Treiber von anderen Herstellern integrieren kann. Auch hier besteht eine zentrale Datei, auch Antwortdatei genannt, welche die Einstellungen für die unbeaufsichtigt Installation enthält. Diese kann mit einem speziellen Tool (Setupmgr.exe) erstellt werden. In dieser Datei in lassen sich allgemeine, Netzwerk und erweiterte Einstellungen vornehmen. Unter den allgemeinen Einstellungen lassen sich z.B. die Benutzerinformationen, Computerfunktionen, Pfade usw. definieren. Im Netzwerkteil lassen sich die Netzwerkkarten , Protokolle, Einstellungen für die Protokolle, Interneteinstellungen und Dienste konfigurieren. Unter den erweiterten Einstellungen lassen sich externe Geräte wie z.B. Tastatur, Anzeige und das Dateisystem auswählen.
Ein typischer Eintrag für die Konfiguration eines Netzwerkprotokolls sähe in dieser Datei z.B. folgendermaßen aus:

```
[TCParamSection]
     DHCP=no
     IPAddress=192.168.0.1
     Subnet=255.255.0.0
     Gateway=0.0.0.0
     DNSServer=192.168.0.100,192.168.0.200
     DNSName=rechner1.de
```

Verwendet man für das Setup eine solche Datei, gibt es keine Möglichkeit dynamische Konfigurationsdaten zu verwenden. Aus diesem Grund können so genannte UDF (Uniqueness Database Files) verwendet werden. Sie haben die

gleiche Struktur, wie die oben beschriebene Datei. Ihre Funktion besteht darin spezielle Abschnitte der Antwortdatei zu ersetzen. Eine solche Datei besteht aus zwei Abschnitten. Zum einen die UniqueIDs, welche die Abschnitte der Antwortdatei identifiziert die ersetzt werden sollen. Zum anderen die UniqueID-Parameter, welche die tatsächlich in der Antwortdatei zu ersetzenden Daten enthält.
Eine solche im UDF-Datei kann z.B. folgendermaßen aussehen:

```
[UniqueIDs]
     Max=Userdata,Network
     Moritz=Userdate,Network

     [Max:Userdate]
          FullName= „Max Mustermann“
          ComputerName= „PC_Max“

     [Moritz:Userdate]
          FullName= „Moritz Mustermann“
          ComputerName= „PC_Moritz“

     [Max:Network]
          InstallAdapters= 3com-III
          IPAddress=192.168.0.2

     [Moritz:Network]
          IPAddress=192.168.0.3
```

Hat man die oben beschriebenen Dateien erstellt, so muss dem Setup Programm beim Programmaufruf nur noch die Namen der Dateien, inklusive diverser Optionen, mitgegeben werden. Daraufhin erfolgt die Installationen vollautomatisch.
Quellen [10] [19] [21]

2.3.2 Suse Linux

Seit der Version 8.0 bietet Suse Linux den so genannten „AutoYast“ Dienst, zur vollautomatischen Installation, an (vorherige Versionen ALICE mit Yast1).
AutoYast ist ein Mechanismus um von einem zentralen Punkt aus eine Installation vollautomatisch durchführen zu lassen. AutoYast unterstützt dabei verschiedene Systemkonfigurationen und Betriebssystemversionen. AutoYast hat den selben Entwicklungszyklus wie Yast2 (Installationsprogramm von Suse Linux, vergleichbar mit dem Windows Installationsprogramm), was zur Folge hat, wenn Yast2 mehr einen grösseren Funktionsumfang bekommt, so erhält auch automatisch AutoYast mehr Möglichkeiten.

Basiskonzept:
Ähnlich wie bei Windows 2000/XP wird auch hier eine Steuerdatei (vergleiche Windows Antwortwortdatei) mit allen benötigten Informationen erzeugt. Diese Datei wird im XML- Format abgespeichert und kann so mit jedem beliebigen Editor erstellt oder modifiziert werden.

Was ist XML ?

1996 begann das W3-Konsortium, an der Extensible Markup Language (XML) zu arbeiten. Als im Herbst 1997 der Internet Explorer 4 herauskam, unterstützte dieser im begrenztem Umfang XML. Anfang 1998 wurde XML zum offiziellen Standard des W3-Konsortiums erhoben und liegt derzeit in der Version 1.0 vor. Der "Chefentwickler" von XML ist der Kanadier Tim Bray. Es dauerte noch knapp ein Jahr, bis mit dem Internet Explorer 5 der erste Web-Browser herauskam, der das standardisierte XML unterstützte. Alle aktuellen Browser unterstützen inzwischen XML [18].

Vorteile von XML:

- In XML besteht die Möglichkeit eigene Tags zu definieren
  z.B. <meinTag>Inhalt</meinTag>
- Zu jedem selbst definierten Tag können auch die Attribute selbst definieren
  z.B. <meinTag farbe=“bunt“>Inhalt</meinTag>

- Es kann eine unendliche Anzahl von Tags definieren -> beliebige Erweiterbarkeit
- XML lässt sich für viele Anwendungen im Internet einsetzen z.B. als Datenbank, als Dokumenten- Management, als Listenverwaltung usw.....
- XML trennt die Daten von der Darstellung. Die Darstellung der Daten erfolgt mittels XSL (Extensible Stylesheet Language). Dadurch sind XML-Dateien für den Betrachter leichter zu verstehen.

Eine solche Steuerdatei kann von Hand erstellt werden. Die komfortablere Möglichkeit für die Erzeugung der für die automatische Installation besteht jedoch darin, die XML- Datei mittels Yast2 zu erzeugen.

Hierzu ist es notwendig, dass die folgenden Pakete installiert sind (SuSe Linux 9.0):

- autoyast2 – Basispaket für das Management der Steuerdatei und die Konfiguration von AutoYast
- autoyast2-installation – Paket dass die automatische Installation, in Abhängigkeit von der Steuerdatei, erzeugt
- autoyast2-utils (optional) – Diverse Utilities für die automatische Installation
- yast2-xml – XML Agent für Yast2

Sind die oben erwähnten Pakete installiert, so erhält man im YaST Control Center unter dem Auswahlpunkt Sonstiges einen zusätzlichen Auswahlpunkt Automatische Installation.

Abbildung 3 - Kontrollzentrum „Automatische Installation“

Nun kann komfortabel mit dieser Auswahlmöglichkeit die Steuerdatei mittels einer grafischen Oberfläche erstellt werden. Die folgende Abbildung zeigt diese Oberfläche. Hier hat man nun die Möglichkeiten der Steuerdatei die zu installierenden Pakete, individuelle Skripte, Hardwarekonfiguration, Benutzer, Systemeinstellungen und das Netzwerk inklusive NFS-, Proxy-, Samba-, NIS-, HTTP-Server zu konfigurieren.

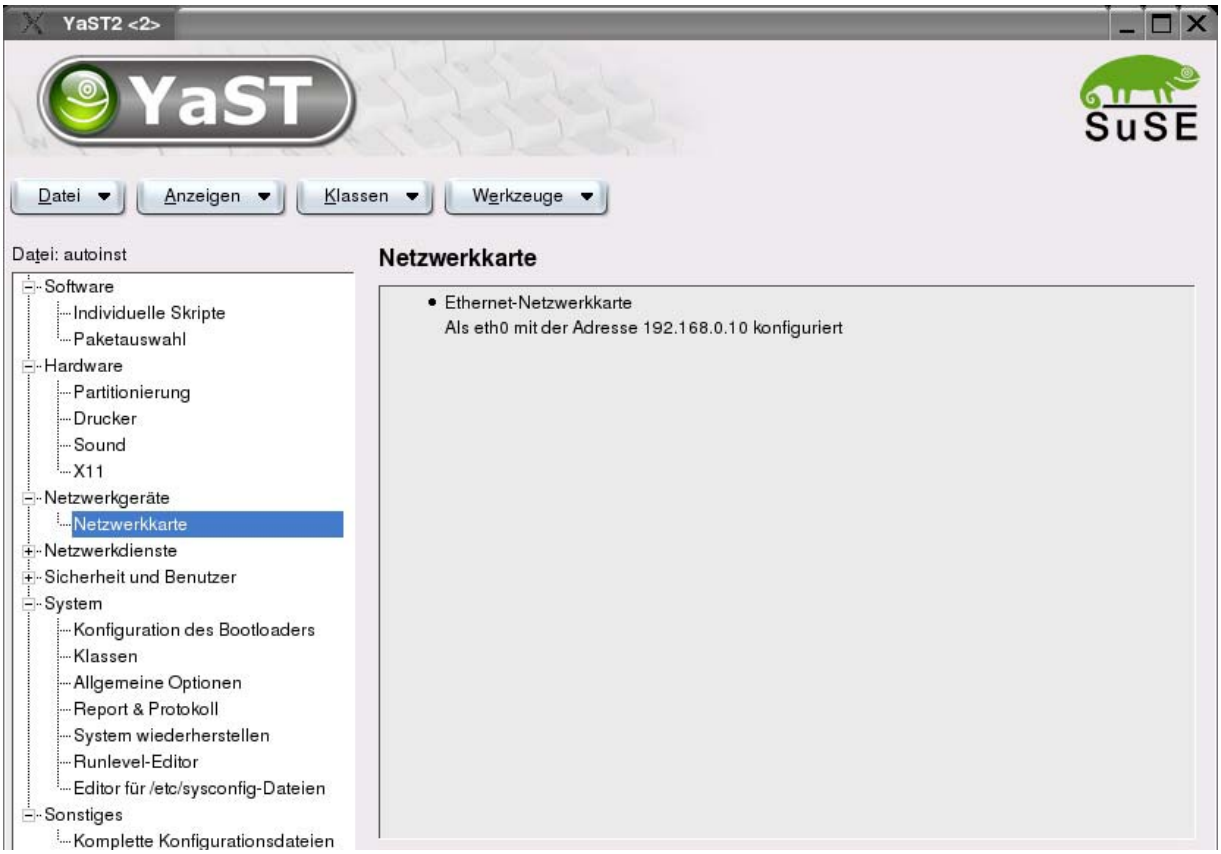


Abbildung 4 - autoinst.xml mit YaST erstellen

Sind alle benötigten Einstellungen gemacht, so muss nur noch mittels Datei -> Speichern unter, die autoinst.xml Datei (Steuerdatei) erzeugt werden. Der folgende Auszug aus dieser Datei zeigt die Konfiguration der Netzwerkkarte inklusive diverser Einstellungen wie z.B. DNS-Server des TCP/IP Protokolls.

```
<networking>
  <dns>
    <dhcp_hostname config:type="boolean">false</dhcp_hostname>
    <dhcp_resolv config:type="boolean">false</dhcp_resolv>
    <domain>abb-wkgproutap</domain>
    <hostname>linux-rechnername</hostname>
    <nameservers config:type="list">
      <nameserver>192.168.0.100</nameserver>
      <nameserver>192.168.0.200</nameserver>
    </nameservers>
  </dns>
  <interfaces config:type="list">
    <interface>
      <bootproto>static</bootproto>
      <broadcast>192.168.0.255</broadcast>
      <device>eth0</device>
```

```
        <ipaddr>192.168.0.10</ipaddr>
        <netmask>255.255.255.0</netmask>
        <network>192.168.0.0</network>
        <startmode>onboot</startmode>
      </interface>
    </interfaces>
    <modules config:type="list">
      <module_entry>
        <device>eth0</device>
        <module>e100</module>
      </module_entry>
    </modules>
    <routing>
      <ip_forward config:type="boolean">false</ip_forward>
      <routes config:type="list">
        <route>
          <destination>default</destination>
          <device>-</device>
          <gateway>192.168.0.1</gateway>
          <netmask>-</netmask>
        </route>
      </routes>
    </routing>
  </networking>
```

Die so erzeugte Datei kann nun mit jedem handelsüblichen Editor, besser geeignet wäre ein XML-Editor, modifiziert werden. Es ist zwingend darauf zu achten, dass die XML-Spezifikation für die Tags eingehalten wird (geöffneter Tag muss auch geschlossen werden), da ansonsten die Installation mit einem Fehler abgebrochen wird. Da die Installation via Netzwerk erfolgen soll, müssen die entsprechenden Installations- CDs der Distribution auf dem Server angelegt und freigegeben werden. Anschliessend muss dem Setup (in der Datei isolinux.cfg) nur noch mitgeteilt werden, dass es sich um eine automatische Installation handelt, und wo die Steuerdatei gefunden werden kann [9].

2.3.3 HP Tru64 Unix

Seit der Version 10.x wird HP-Unix mit dem so genannten Ignite-UX ausgeliefert. Dabei handelt es sich um einen Mechanismus, zur automatischen Installation von Clients. Das Ignite Modell basiert auf einem typischen Client- Server Modell. Dies hat die Vorteile, dass mehrere Clients gleichzeitig installiert werden können, spezielle

Anpassungen (Konfigurationen) gemacht werden können und dass der Status der Installation zentral überprüft werden kann. Vorraussetzung hierfür ist jedoch, dass ein HP-Unix Server im Netzwerk existiert, auf welchem die entsprechende Version (abhängig von der Betriebssystemversion) des Ignite Pakets installiert ist. Weiterhin unterstützt Ignite das Wiederaufsetzten von Rechnern (Disaster Recovery) und das Updaten bestehender Betriebsysteme.
Auch hier besteht die Möglichkeit, mittels eines Konfigurationsfiles spezielle Anpassungen (z.B. Partitionierung, zu installierende Software, auszuführende Skripte, Hardware) für die Installation zu definieren. Ein so erzeugtes Konfigurationsfile kann anschliessend von Hand bearbeitet werden oder mittels Abfragen automatisiert werden. Eine solche Abfrage könnte zum Beispiel folgendermassen aussehen (C ähnliche Grammatik) :

```
(memory > 512Mb)
{
      init _hp_pri_swap=512Mb
}
else
{
      init _hp_pri_swap=memory*2
}
```

Hierbei wird überprüft ob der zu installierende Client einen Arbeitsspeicher grösser 512 MB besitz. Ist dies der Fall, so wird die Grösse der Swap-Partition auf 512 MB gesetzt. Anderenfalls auf die zweifach Arbeitsspeichergrösse.

Weiterhin besteht die Möglichkeit ein in der Ignite Welt so genanntes „Golden System“ zu generieren. Darunter versteht man eine hardwareunabhängige und konfigurierbare automatische Installation, die mittels zur Laufzeit auf dem Server erzeugten Konfigurationsdateien und Installationsskripten erfolgt.

Weitere Möglichkeiten zur vollautomatischen Installation bestehen durch Verwendung spezieller Software wie z.B. InstallAnywhere der Firma ZeroG. Weitere Informationen können auf der Homepage www.zerog.com bezogen werden [20].

2.3 Anwendungen

In den folgenden Unterpunkten werden die Möglichkeiten zur vollautomatischen Anwendungsinstallation unter Windows und Linux/UNIX näher beschrieben.

2.3.1 Möglichkeiten unter Windows (MSI)

Unter Windows besteht eine Möglichkeit für eine vollautomatische Installation von Anwendungen, sofern diese ein MSI File enthalten. Die Bezeichnung MSI steht für Microsoft Installer. Dabei handelt es sich um eine Datenbank, die die angezeigten Setup Fenstern, kopierenden/ändernden Dateien, Registry Einträge und alle Anweisungen für Installation, Update und Deinstallation der Anwendung enthält. Ein MSI-File ist an der Dateiendung msi zu erkennen (z.B. office10.msi). Zusätzlich existiert meist noch eine exe-Datei, meist Setup.exe, deren Funktion es lediglich ist, den Installer aufzurufen.
Der Installer kann mittels msiexec aufgerufen werden. Hierbei ist es möglich diesem verschiedene Parameter zur Konfiguration mitzugeben. Anschliessend folgt eine Aufstellung der wichtigsten Parameter sowie deren Aufruf bzw. weiterreichen an den Microsoft Installer [10].

Wichtigsten Optionen (Switches) des Microsoft Installer :

```
/i Package|ProductCode Installiert oder konfiguriert ein Produkt.
/f Package|ProductCode Repariert ein Produkt.
/a PackageAdministrative Installationsoption Installiert ein Produkt
   im Netzwerk
/x Package|ProductCode Deinstalliert ein Produkt
/l Logfile Spezifiziert den Pfad der Protokolldatei
/p Patch/Package Wendet ein Patch an
/q Legt die Benutzeroberflächen-Ebene fest
```

```
Msiexec <Option> <Pfad zum MSI File>
```

Des Weiteren ist es möglich dem Installer so genannte Transformationsfiles mitzugeben. Diese Files können anhand der Dateiendung .mst identifiziert werden. In diesen Files können verschiedene Konfigurationen gespeichert werden wie z.B. Pfadangaben, Seriennummer usw.. Mit diesen Einstellungen ist es möglich Teile oder sogar das ganze Setup, ähnlich dem „unattended Setup“ von Windows zu konfigurieren bzw. zu automatisieren. Das Transformationsfile macht nichts anderes, als die Datenbank (siehe MSI) abzuändern. Dies geschieht zur Laufzeit des Setups. Somit ist es nicht möglich eine solche Transformation auf eine bereits installierte Software nachträglich anzuwenden.
Ein solches Transformationsfile muss mit speziellen Programmen erzeugt werden und kann daher auch nicht mit einem normalen Texteditor betrachtet werden.

Der Aufruf des Installers unter Verwendung eines Transformationsfiles (trans.mst) am Beispiel Microsoft Office sähe z.B. folgendermassen aus:

```
Msiexec /i D:\Setups\MSOffice10\office10.msi TRANSFORMS=trans.mst
```

2.3.2 Möglichkeiten Linux/UNIX

Das Pendant der Windows MSI Datei sind unter Linux bzw. UNIX die so genannten RPM Dateien. "RPM" ist die Abkürzung für Red Hat Package Manager und kennzeichnet Programmarchive, die nach dem von Red Hat eingeführten und heute von fast allen Distributionen verwendeten Standard zusammengestellt werden.
Unterschieden wird dabei zwischen sogenannten Binär-RPMs (kompilierte Version) und Source-RPMs (Quelltexte die erst noch kompiliert werden müssen). Binär-RPM-Pakete enthalten die ausführbaren Dateien sowie alle weiteren Konfigurations- und sonstigen Dateien, die Teil des Programms sind. Neben diesen Dateien bietet ein RPM-Archiv zusätzlich Informationen darüber, was unmittelbar vor und nach der Installation zu tun ist. Zusätzlich ist dort noch vermerkt, welche weiteren RPM-Pakete für die Installation benötigt werden oder mit welchen anderen Paketen es zu (Versions-) Konflikten kommen kann.

Diese RPM-Pakete lassen sich in der Regel sehr einfach durch die Eingabe eines einzigen Befehls oder über ein grafisches Tool (z.B. kpackage bei SUSE Linux) installieren. Solche Pakete erkennt man an der Endung .rpm. Für die Installation solcher RPM-Pakete werden Systemadministratorrechte (root) benötigt. Aufgrund der Unterschiede zwischen den einzelnen Linux- Distributionen, existieren häufig verschiedene Pakete für die entsprechenden Distributionen. Am Dateinamen eines Pakets kann in der Regel erkannt werden, um welche Version des Programms es sich handelt, die Prozessorplattform und eventuell für welche Distribution das Paket erstellt wurde.

Beispiel: `PaketXY-1.0-SuseLinux.i386.rpm`

Aus obigem Beispiel lässt sich der Paketname (PaketXY), die Versionsnummer (1.0), die Distribution (Suse Linux) und die Prozessorplattform (Intel 386) erkennen.

Zur vollautomatischen Installation eignet sich, anstatt der grafischen Oberfläche, das Dienstprogramm rpm jedoch besser. Dieses Programm kann direkt in einem Terminal- Fenster gestartet werden. Alles was zum Ausführen einer Installation nötig ist, ist der genaue Dateiname inklusive Pfad des RPM- Pakets und Administratorrechte. Mit folgendem Aufruf könnte die Installation im Terminal-Fenster erfolgen:

```
rpm -Uvh pfad/PaketXY-1.0-SuseLinux.i386.rpm
```

Wurde die Installation auf diese Weise gestartet, so zeigt ein Fortschrittsbalken an den Installationsstatus des Pakets an.

Die zur Verfügung stehenden Optionen sind in folgender Tabelle zusammengefasst:

-i	Installieren (kein Update)
-U	Update (oder auch Erstinstallation)
-v	"verbose" (ausführlich); zeigt Paketnamen an
-H	zeigt Fortschrittsbalken an

-q	"query" (Abfrage); nachfragen, ob ein Paket installiert ist
-e	"erase": Löschen eines Paketes
--nodeps	Abhängigkeiten ignorieren (also auch bei fehlenden Paketen installieren)
--force	Installation erzwingen, auch bei Konflikten

Tabelle 3 - RPM Installationsoptionen

Diverse Linux/UNIX Distributionen (z.B. HP True64 Unix) bieten des Weiteren die Möglichkeit für die zu installierende Anwendung auf einem Referenzsystem das Setup zu durchlaufen und daraus eine Datei zu generieren. Das Prinzip welches dahinter steckt entspricht im Wesentlichen der Unattended- Installation bei Windows, bzw. AutoYast2 bei Linux. Daher wird auf diese Methode nicht weiter eingegangen.
Quellen [5][6][7][8]

2.4 Systeme zur automatischen Softwareverteilung

In folgendem Abschnitt wird auf verschiedene Deployment Programme eingegangen. Zugrunde liegend hierfür ist ein Artikel aus der Computerzeitschrift PC-Professional Ausgabe 06/2003. Diese Programme führen eine vollautomatische Betriebssysteminstallation, als auch die Verteilung von Software in Netzwerken durch. Alle im Vergleich angesprochenen Programme erlauben eine zentrale Steuerung von einem abgesetzten Server. Dabei kommen 2 Prinzipien der Softwareverteilung zum Einsatz. Zum einen die Push-Methode, bei dem der Server die Software an die Clients verteilt, zum anderen die Pull-Methode, bei der sich der Client die Software vom Server holt.

Testkandidaten:

- Deployment Solution 5.5 (Hersteller Altiris)
- Empirium Pro 9.0 (Hersteller Matrix42)

- Enterprise Managment Suite 5.1 (Hersteller Baramundi)
- Landesk 6.62 (Hersteller Landesk)
- Netinstall 5.56 (Hersteller Netsupport)
- OnCommand CCM 5.4 (Hersteller OnTechnology)
- SMS 2.0 (Microsoft)
- Zenworks 4.0 (Novell)

Als Testumgebung wurde ein typisches Szenario eines mittelständischen Unternehmens simuliert. In einem solchen Unternehmen werden meist eine Vielzahl unterschiedlich ausgestatteter Computer eingesetzt. Daher wurden Rechner mit unterschiedlicher Hardware bei diesem Test eingesetzt. Die Palette reicht dabei vom Pentium II/200 über Pentium III bis zum Pentium IV, Hauptspeichergrösse von 128 oder 256 MByte und unterschiedlichen Chipsätzen, Grafikkarten, Netzwerkkarten und Festplatten.

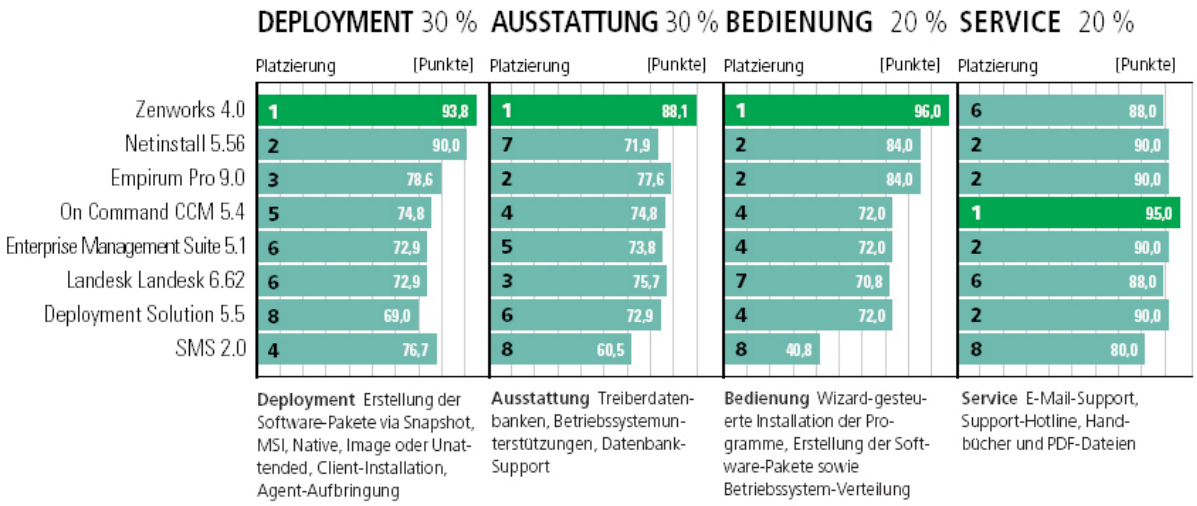


Abbildung 5 - PC Professional Testergebnis Details

In diesem Test landet OnCommand CCM „nur“ auf dem vierten Rang. Im Testergebnis wird das System zwar als mächtiges Tool zur Softwareverteilung beschrieben, jedoch aufgrund der sehr komplexen Bedienung, sowie der nicht so komfortablen Erzeugung der Skripte bzw. der Wizards, sowie einiger fehlender Möglichkeiten (wie (z.B. Warnmeldungen via SNMP) im landet es im Vergleich mit den anderen System hinter diesen.

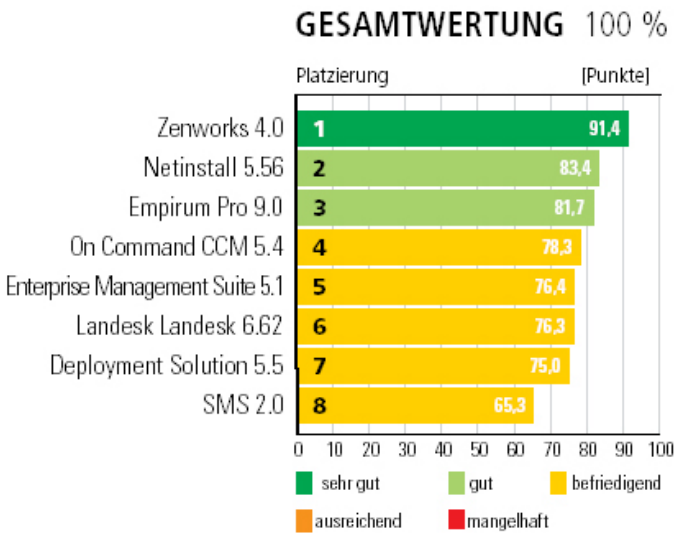


Abbildung 6 - PC Professional Gesamt

Absoluter Spitzenreiter in diesem Vergleich wurde Zenworks 4.0 von Novell, das in allen Kategorien (ausser Service) die Spitzenposition einnahm. Absoluter Verlierer diese Tests ist jedoch Microsoft System Management Server 2.0. Dies ist sicherlich darauf zurückzuführen, dass das Programm seit seiner Einführung im 1998 nahezu unverändert blieb. Dabei wurden die unkomfortable Bedienung und Konfiguration, Vorhandensein eines Microsoft SQL-Server (andere werden nicht unterstützt), fehlende direkte Betriebssysteminstallation (RIS benötigt) und die erforderliche Domäne bemängelt.

Das komplette Ergebnis des Vergleichs, kann im Anhang F – Testbericht PC-Professional 06/2003 eingesehen werden [12].

2.5 Verwendete System

In folgendem Kapitel wird auf das verwendete System, die Architektur, die Firma und Produkte genauer eingegangen.

2.5.1 Über OnTechnology

Die Firma OnTechnology Europe wurde 1988 in Starnberg unter dem Namen CSD GmbH gegründet. Seid 1997 ist sie eine 100% Tochter der ON Technology mit ihrem Firmensitz in Waltham Massachusetts / USA. Weltweit beschäftigt die Firma ca. 180 Mitarbeiter, wovon ein grosser Anteil (ca. 70 Mitarbeiter), in Starnberg beschäftigt sind. Aufgrund des grossen Erfolgs der Firma mit ihrem CCM System, wurde sie Anfang 2004 von Symantec übernommen.

OnTechnology ist ein führender Anbieter von Infrastruktur Management Lösungen. Unter Infrastruktur Management versteht man die Verwaltung und Überwachung der bestehenden IT Infrastruktur. Dies beinhaltet die Aspekte Sicherheit, Installation, Update, Wartung, Konfiguration, Performance und Skalierbarkeit [17].

2.5.2 Produkte

On Command CCM

Zur Reduzierung der Kosten für die IT-Infrastruktur (siehe auch Punkt 1.1) hält die Firma OnTechnology ihr Produkt On Command CCM bereit. Der Begriff CCM steht dabei für Comprehensive Client Management was so viel wie komplette Clientverwaltung bedeutet. Mit dieser Lösung ist es möglich, vollautomatisch und von einer einzigen Konsole aus, neue Hardware (softwaremässig) zu installieren, Migration auf neue Betriebssysteme, De- und Installation sowie Konfiguration von Applikationen, Verteilung von Patches, Updates und Service Packs zu integrieren. Ein weiterer wichtiger Aspekt ist sicherlich auch Disaster Recovery. Auch dies ist mittels des Systems möglich. So kann ein Client mittels eines Mausklicks in seinen ursprünglichen Zustand versetzt werden. Des weiteren wird der ganze Installations- oder Updateprozess mitprotokolliert und dem Benutzer ständig der aktuelle Stand des Prozesses angezeigt.

On iCommand

Mittels dem Produkt iCommand, besteht die Möglichkeit, mobile User sowie Linux Systeme über das Internet zu managen. Hierbei bestehen die gleichen Möglichkeiten wie beim On Command CCM.

On Command Remote

Dieses Produkt dient zur zentralen Diagnose, Problemlösung und Überwachung von kritischen Computern (z.B. Servern). Der Benutzer enthält dabei die volle Tastatur und Mauskontrolle des Computers. Dieses Produkt kann im Leistungsumfang mit Remotelösungen wie z.B. Symantec PcAnywhere verglichen werden.

On Command Discovery

Mittels dieses Produkts ist es möglich, die vorhandenen IT-Werte (Hardware und Software) zu inventarisieren. Das Produkt bietet die Möglichkeit der Hardware- und Softwareerfassung, die Ermittlung des physischen Standorts (mittels „LANProbe“), die Messung der Softwarenutzung und wann Änderungen an diesen vorgenommen worden sind.

2.5.3 Technology des Systems

Der Hauptbestandteil des CCM-Systems ist der Sybase Adaptive Server Anywhere. Hierbei handelt es sich um ein hochperformantes relationales Datenbankmanagementsystem. In dieser Datenbank werden alle CCM Informationen gespeichert, einschließlich der Administratoraccounts, verwaltenden Computer, Computergruppen, CCM Pakete, Paketprofile und zeitgesteuerten Aufträge.

Für die Kommunikation mit der Datenbank und den Clients, kommen in Java programmierte Anwendungen zum Einsatz. Dies hat den Vorteil, dass das System Plattformunabhängig eingesetzt werden kann. Einzige Vorraussetzung hierfür, ist eine installierte Java Runtime Environment (JRE) bzw. Virtuelle Maschine von SUN.
Der Datenaustausch von Client- und Server-Rechner erfolgt mittels RMI.
Quellen [13][14][15][16][17]

Was versteht man unter RMI ?

Hinter dem Begriff RMI steht „Remote Method Invocation“. Hierbei handelt es sich um einen Java spezifischen Mechanismus, der es ermöglicht, auf einfache Art und Weise Objekte im Netzwerk zu verteilen und ihre Dienste anderen Arbeitsplätzen zugänglich zu machen. Hierdurch erreicht man eine einfache Art der verteilten Anwendungsprogrammierung, da sich der Programmierer nicht intensiv mit der Netzprogrammierung auseinandersetzen muss. RMI ist jedoch ein höher entwickelter Mechanismus zur Kommunikation als eine einfache Socket-Verbindung. Vorraussetzung hierfür ist natürlich wieder, dass auf dem Client sowie dem Server eine JRE (ab Version 1.1) installiert ist.

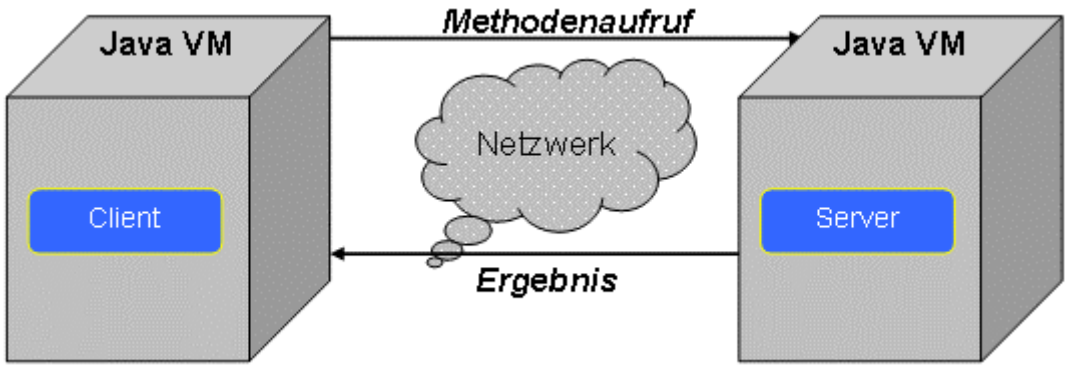


Abbildung 7 - Remote Method Invocation

Die Architektur von RMI besteht aus 4 Komponenten. Eine dieser Komponenten ist der Server, welcher die Remote Dienste bereitstellt. Eine weitere ist der Client, der die vom Server zur Verfügung gestellten Dienste in Anspruch nehmen möchte. Des Weiteren existiert eine Registry, mit deren Hilfe die Identifizierung der Objekte bzw. zur Verfügung gestellten Dienste erfolgt. Dazu müssen diese Serverobjekte bei diesem angemeldet werden, um daraufhin für einen Client erreichbar zu sein. Als letztes die Kommunikationskomponente, die wie der Name schon ahnen lässt, für die Kommunikation der anderen drei Komponenten verantwortlich ist. Durch diese Architektur ist es möglich mit den Methoden des Servers wie mit lokalen Methoden zu arbeiten.

Die prinzipielle Arbeitsweise von RMI lässt sich wie folgt erklären:

- Auf dem Server werden in einem *Remote-Interface* die zur Verfügung gestellten Methoden definiert, die von den Clients aufgerufen werden können
- Die Serverklasse implementiert dieses Interface und erzeugt daraus die Instanz(en). Diese werden als *Remote-Objekte* bezeichnet.
- Diese Objekte müssen dann in der RMI-Registry registriert werden. Mit Hilfe des darin enthaltenen Name-Service können die Clients mittels Namen auf diese Objekte zugreifen.
- Sind die vorherigen Schritte erledigt, so kann sich ein Client mittels der RMI-Registry eine Referenz, *sogenannte Remote-Referenz*, auf das benötigte Objekt beschaffen und daraufhin die vom Objekt zur Verfügung gestellten Methoden ausführen. Die Ausführung dieser Methoden erfolgt hierbei jedoch auf der Serverseite. Der Rückgabewert der Methode wird dann nach der Ausführung an den Client zurückgesendet.

Anmerkung: Die Vorgehensweise und Architektur von RMI ist mit CORBA (Common Object Request Broker) vergleichbar

Quellen [1][2]

2.5.4 Vorteile des Systems

Einer der größten Vorteile des Systems ist sicherlich, dass es im Gegensatz zu anderen Systemen direkt nach dem starten BIOS des Computers eingreift. In folgender Zeichnung soll dies ein wenig verdeutlicht werden.

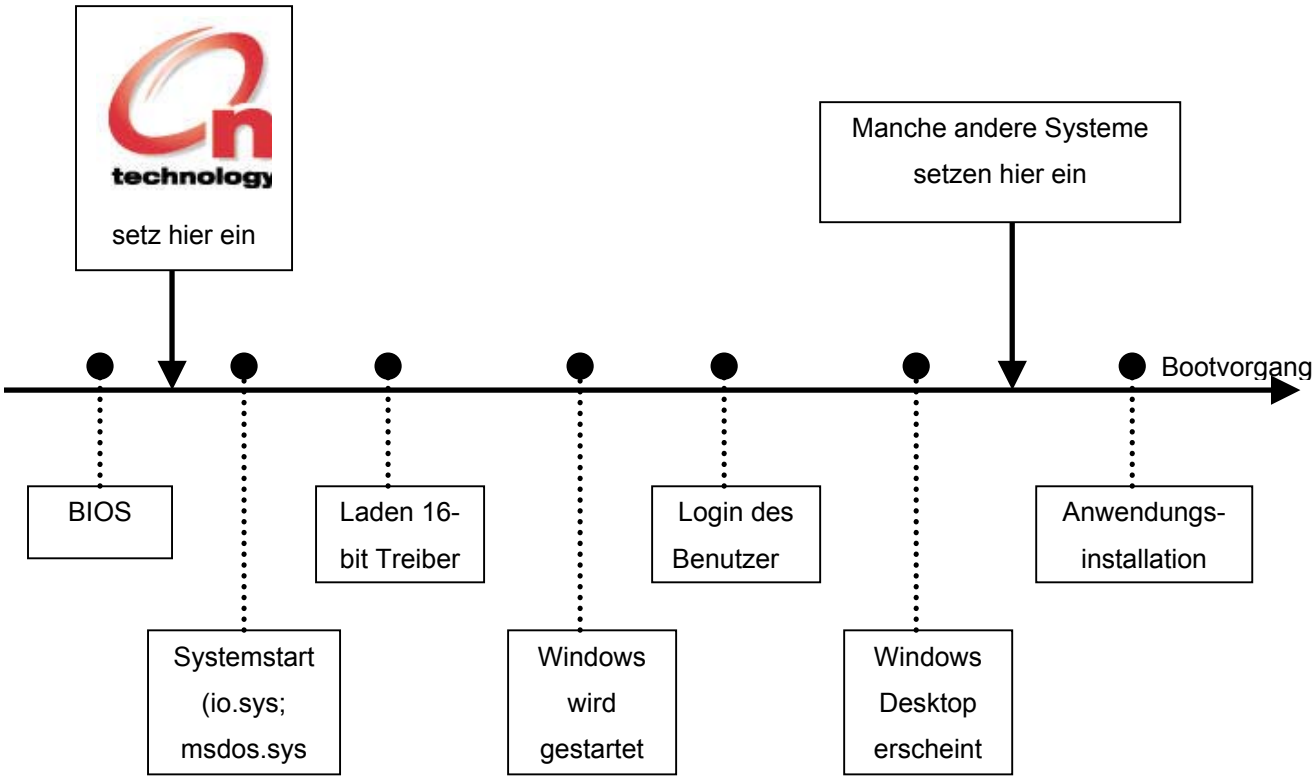


Abbildung 8 - Wo startet CCM ?

Somit hat man bereits nach der BIOS-Routine die volle Kontrolle über den Client und kann diesem an diesem Punkt schon mitteilen was zu tun ist.
Weiterer Vorteil des Systems in Punkto Sicherheit ist, dass die Installation mittels spezieller Administrator-Rollen und Passwortschutz erfolgt. Durch das Definieren von Abhängigkeiten der einzelnen Pakete können Fehler vermieden, sowie die Installationsreihenfolge der Pakete festgelegt werden. Jeder Client kann individuell (z.B. mit speziellen Lizenzkeys oder Partitionen) konfiguriert und aufgesetzt werden. Durch die Möglichkeit zur Definition von Client-Profilen, mittels denen man dann dynamische oder statische Gruppen bilden kann, ist auch die Arbeit mit einer grossen Anzahl von Clients kein Problem mehr. Der gemanagte Client ist während des Vorgangs vor Eingriffen durch den User komplett gesperrt (Maus & Tastatur ohne Funktion).

Ein weiterer grosser Vorteil des Systems ist sicherlich, dass es im Gegensatz zu vielen anderen Systemen, nicht mittels Snapshot- Technologie arbeitet. Bei der Snapshot Technologie wird ein Musterrechner vor und nach der Installation der

gewünschten Software untersucht, und aus den Unterschieden ein Differenzpaket gebildet, welches später dann auf beliebige Clients im Netzwerk verteilt wird. Bei dieser Vorgehensweise landet man evtl. in der Windows DLL „Hölle“. Ist eine benötigte DLL-Datei z.B. schon auf dem Muster System vorhanden, so wird dies von der Snapshot- Technologie nicht erkannt. Somit gelangt diese DLL auch nicht in das Differenzpaket. Dies kann zur Folge haben, dass ein eingespieltes Softwarepaket nicht korrekt, oder sogar gar nicht, auf dem Zielsystem funktioniert. Außerdem muss die Installation auf allen zum Einsatz kommenden Betriebssystemen durchgeführt werden, da je nach System unterschiedliche Dateien auf den Rechner kopiert werden. Ein weiterer Nachteil dieser Methode ist, dass Softwarehersteller in der Regel keinen Support leisten wenn ihre Programme nicht "ordnungsgemäß“ installiert worden sind. Das System der Firma OnTechnology geht hierbei einen anderen Weg. Es installiert die Software durch vorantreiben des Originalsetups. Somit können die oben genannten Probleme nicht entstehen.

Quellen [13][14][15][16][17]

3. Umsetzung

In diesem Kapitel wird der Systemaufbau, die Systemkonfiguration, die Skripterstellung sowie die aufgetretenen Probleme erläutert.

3.1 Systemaufbau

In folgenden Unterpunkten wird auf den Server, das Raid und den Systemaufbau genauer eingegangen.

3.1.1 Server

Bei dem eingesetzten Server handelt es sich um ein Gerät der Firma Transtec aus der Modellreihe 2600W. Dieses Modell besteht aus einem rack-mountable Gehäuse mit folgender Spezifikation.

- Intel E7501 Chipsatz
- Bis zu 7 Hot Swap Platten
- Onboard dual Intel 10/100/1000BASE-T
- Onboard Adaptec Dual Channel AIC-7902W U320
- 3 PCI-X 64-bit, 100 MHz
- 3 PCI 64-bit, 66 MHz
- Triple redundant 500W Netzteil

Als Prozessoren kommen dabei zwei INTEL Xeon Prozessoren mit jeweils 2,4 GHz zum Einsatz. Der Hauptspeicher hat eine Grösse von 1 GB ECC Ram. Die internen Festplatten im Server wurden entfernt. Stattdessen kommt als Datenspeicher ein externes RAID zum Einsatz (siehe Punkt 3.1.2 RAID)

3.1.2 Raid

Die Abkürzung "RAID" steht für "**Redundant Arrays of Inexpensive Disks**" und bezeichnet Techniken, die mehrere kostengünstige Festplatten zur Speicherung von Daten verwenden, um die Datensicherheit und/oder den Datendurchsatz zu erhöhen. RAID-Systeme kommen besonders in sicherheitskritischen oder durchsatzstarken Datenanwendungen zum Einsatz. Da es verschiedene Möglichkeiten gibt, die Daten redundant zu speichern, werden verschiedene "Raid-Levels" unterschieden.

Die Frage nach dem richtigen Raid-Level für eine Anwendung kann nicht generell beantwortet werden sondern hängt von vielen Faktoren ab.

- Ist Fehlertoleranz wichtig?
- Wie wichtig ist die Geschwindigkeit beim Schreiben/Lesen?
- Kostenfrage bei der Anzahl der Festplatten / Controller.
- Werden häufig kleine Datenmengen (einzelne Blöcke) übertragen?
- Darf die Fehlerwiederherstellung Zeit kosten oder muss dies auch "on the fly" möglich sein?

Im Folgenden werden die verschiedenen Hardware Raid Konfigurationen kurz erläutert [11]. Die Kapazität N bezieht sich dabei auf die Anzahl der vorhandenen Platten.

Raid 0:

Beim Raid 0, dem so genannten „Disk Striping“, werden die Daten in Blöcke aufgeteilt und auf mehrere Platten verteilt.

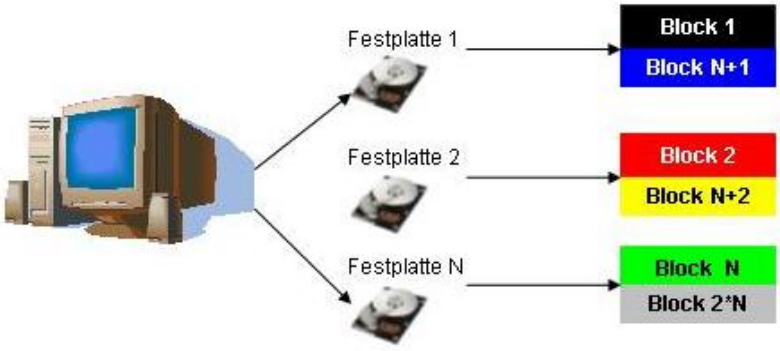


Abbildung 9 - Raid Level 0

Minimale Anzahl benötigter Platten: 2

Kapazität: N

Redundanz: Nein

Vorteile:

- Schnelles Lesen und Schreiben der Daten von den Platten
- Bietet die höchste Performance der Raid Konfigurationen
- Durch das zusammenfassen zu einer einzigen Konfiguration, wird eine bessere Speicherplatznutzung ermöglicht
- Einfaches Design

Nachteile:

- Keine Redundanz und damit auch keine Fehlertoleranz. Beim Ausfall einer Platte gehen alle Daten verloren.

Raid 1:

Der Inhalt einer Festplatte wird auf eine zweite Platte "gespiegelt".

Minimale Anzahl benötigter Platten: 2

Kapazität: N/2

Redundanz: Ja

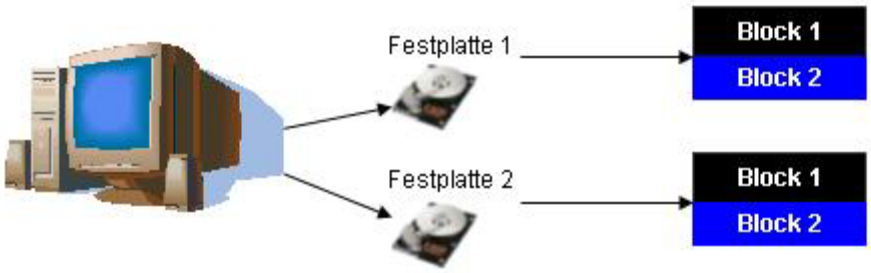


Abbildung 10 - Raid Level 1

Vorteile:

- Schnelles Lesen der Daten von 2 Platten.
- Bei Plattenfehlern werden die Daten einfach von der 2ten Platte wieder zurückgespiegelt.
- Einfaches Design

Nachteile:

- Keine Geschwindigkeitssteigerung beim Schreiben der Daten.
- 100 % Disk-Overhead. (Für z.B. 50GB Speicherplatz müssen 100GB gekauft werden)

Raid 2:

Beim Raid 2, dem so genannten „Disk Striping with ECC“, werden die Daten in Blöcke (8 Bit) aufgeteilt und auf mehrere Platten verteilt. Zusätzlich wird ein Error Correction Code (ECC) mit einer Größe von 2 Bit gebildet und abgespeichert.

Minimale Anzahl benötigter Platten: 10
Kapazität: max. 80 %
Redundanz: Ja

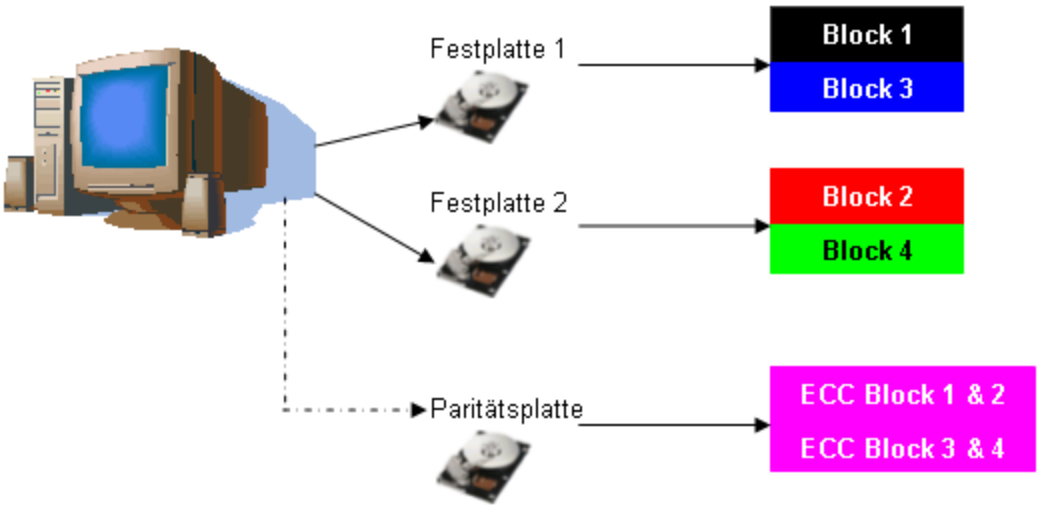


Abbildung 11 - Raid Level 2

Vorteile :

- Hohe Datensicherheit
- Bitfehler bzw. inkonsistente Daten können korrigiert werden
- Hohe Lesegeschwindigkeit (bis 8-fach)

Nachteile:

- Performanceverlust beim Schreiben, durch die Bildung des ECC (Error Correction Code)
- Grösserer Speicherplatzverlust durch Speicherplatzbedarf für den ECC
- Grosser Aufwand für die Implementierung
- Hohe Kosten für die Hardware

Raid 3:

Beim Raid 3, dem so genannten „Disk Striping with Parity“, werden die Daten in Blöcke aufgeteilt und auf mehrere Platten verteilt. Zusätzlich werden auf einer separaten Platte Paritätsinformationen mitgeschrieben, so dass beim Ausfall einer Platte Datenintegrität gewährleistet ist. Die Paritätsinformation wird dabei aus dem ECC (Error Correction Code) erzeugt.

Minimale Anzahl benötigter Platten: 3

Kapazität: N-1

Redundanz: Ja

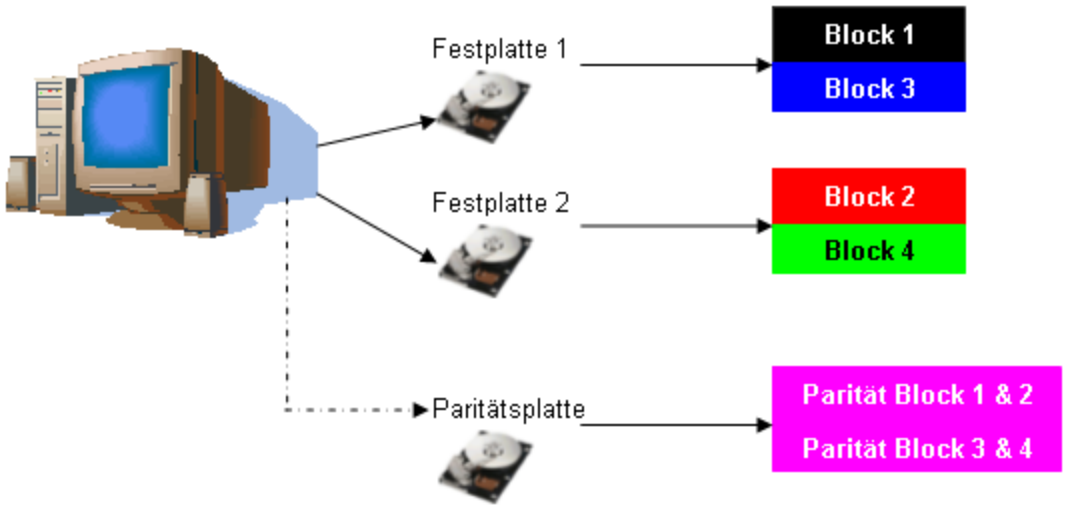


Abbildung 12 - Raid Level 3

Vorteile:

- Hohe Datensicherheit. Datenverlust erfolgt erst beim Ausfall einer Platte und der Paritätsplatte
- Eignet sich hervorragend für Transaktionsverarbeitung, bei denen sich die Plattenoperationen normalerweise aus vielen kleinen Schreib-/Lesetransaktionen zusammensetzt.

Nachteile:

- Performanceverlust beim Schreiben, durch die Berechnung der Parität aus dem ECC (Error Correction Code)

Raid 4:

Beim Raid 4, werden die Daten nicht wie beim Level 3 in Byte-Blöcke aufgeteilt, sondern die kompletten Datenblöcke auf einer Platte gespeichert. Zusätzlich werden

auf einer separaten Platte Paritätsinformationen mitgeschrieben, so dass beim Ausfall einer Platte Datenintegrität gewährleistet ist.

Minimale Anzahl benötigter Platten: 3

Kapazität: N-1

Redundanz: Ja

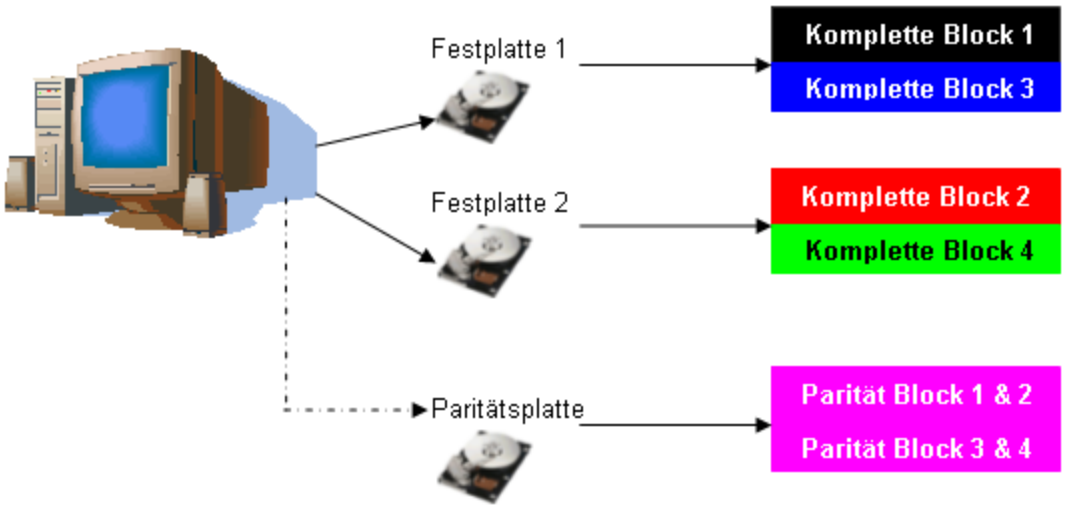


Abbildung 13 - Raid Level 4

Vorteile:

- Hohe Datensicherheit. Datenverlust erfolgt erst beim Ausfall einer Platte und der Paritätsplatte
- Gut geeignet für Operationen mit grossen Datenblöcken
- Gut geeignet in Umgebungen in welchen erheblich mehr Lese- als Schreiboperationen erfolgen

Nachteile:

- Performanceverlust beim Schreiben, da die Paritätsinformationen bei diesem Vorgang gelesen und aktualisiert werden müssen
- Schlechte Eignung für transaktionsoriente Verarbeitung (kleine Transaktionen)

Raid 5:

Daten werden Blockweise auf mehrere Festplatten (mind. 3 Stk.) verteilt, eine Prüfsumme generiert und ebenfalls auf die nächste freie Festplatte geschrieben. Jede Platte hält somit Daten, als auch Prüfsummen.

Minimale Anzahl benötigter Platten: 3
Kapazität: N-1
Redundanz: Ja

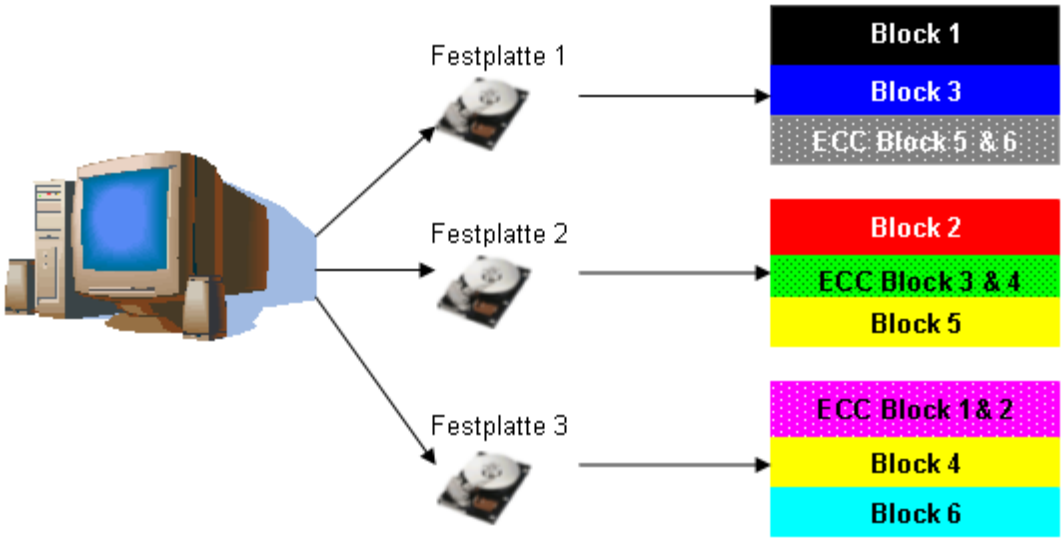


Abbildung 14 - Raid Level 5

Vorteile:

- Sehr hohe Lesegeschwindigkeit
- Gute Schreibgeschwindigkeit
- Gutes Verhältnis von Daten und Prüfsummen

Nachteile:

- Komplexer Aufbau - Teure Controller
- Keine Beschleunigung beim Lesen eines einzelnen Blocks.

Die Raid-Levels 2, 3 und 4 werden aufgrund ihrer oben genannten Nachteile kaum eingesetzt. Des Weiteren gibt es Kombinationen der verschiedenen Raid Levels wie z.B. Raid 10, Raid 50. Unter einem Raid 10 zum Beispiel, auch als Raid 0+1

bezeichnet, versteht man die Kombination aus einem Raid 0 (Striping) und einem Raid 1 (Mirroring). Dadurch ist es möglich diverse Nachteile des einen Raid-Levels durch die Vorteile des anderen Raid-Levels zu kompensieren.

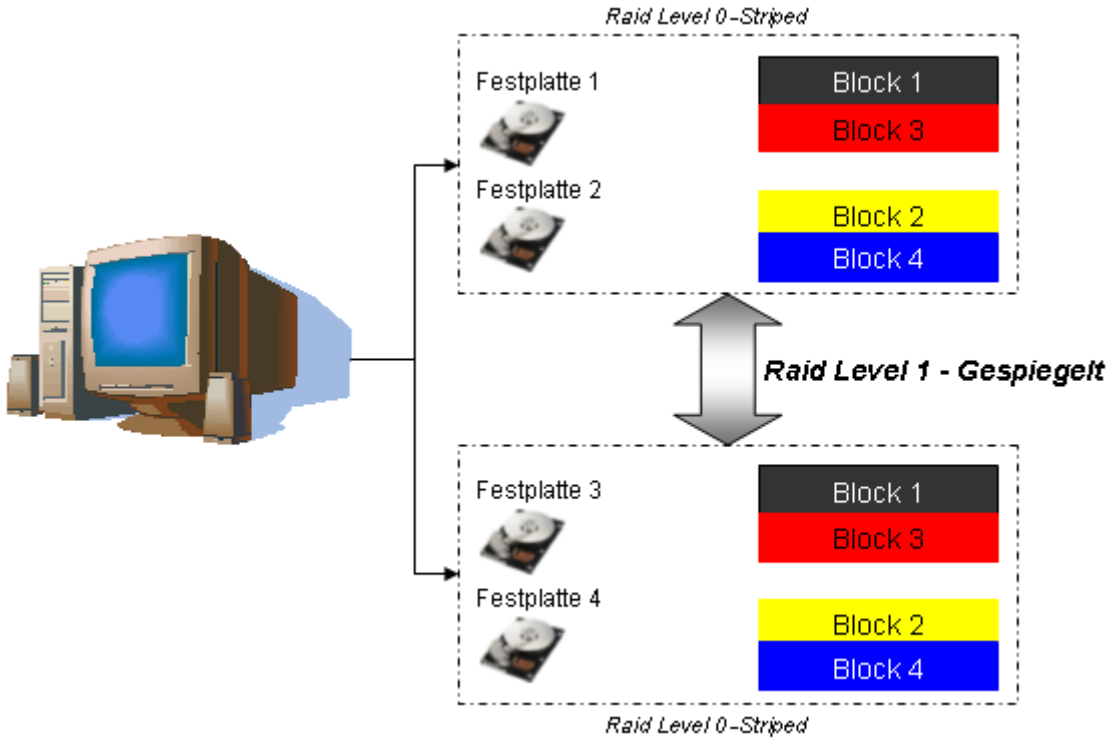


Abbildung 15 - Raid Level 10

Bei dem eingesetzten Raid, handelt es sich um ein 7 Platten Hardware RAID System, der Firma Transtec Modell 5100. Dieses Raid System hat folgende technischen Daten

Technische Daten:

- RAID Level 0, 1, 0+1, 3, 5, 10, 30, 50, JBOD
- 7 Platten hot swap (während des Betriebs austauschbar)
- 2x Ultra 160 SCSI zu den Platten,
- Ultra 160 SCSI LVD zum Host,
- redundante Netzteile und Lüfter

Die Abkürzung JBOD steht für „Just a Bunch of Drives“. Der Raid-Controller behandelt bei dieser Konfiguration jede Platte als normale stand-alone Platte. Jede Platte ist daher eine unabhängige logische Platte. JBOD unterstützt keine Datenredundanz.

Nach dem Vergleich der verschiedenen Raid-Levels und den unterschiedlichen Vor- und Nachteilen, wurde das RAID folgendermassen konfiguriert. Ein RAID 1 (gespiegelt) für die Systemdateien (Windows) bestehend aus zwei 70 GB Festplatten, sowie ein RAID 5 (stripped) bestehend aus vier 70 GB Festplatten (Daten). Die letzte noch nicht verwendete Festplatte wurde als „Global Spare“ definiert. Diese kommt dann zum Einsatz, wenn irgendeine der anderen Platten ausfallen sollte. Passiert dies übernimmt die Platte die Funktion der ausgefallenen Platte.

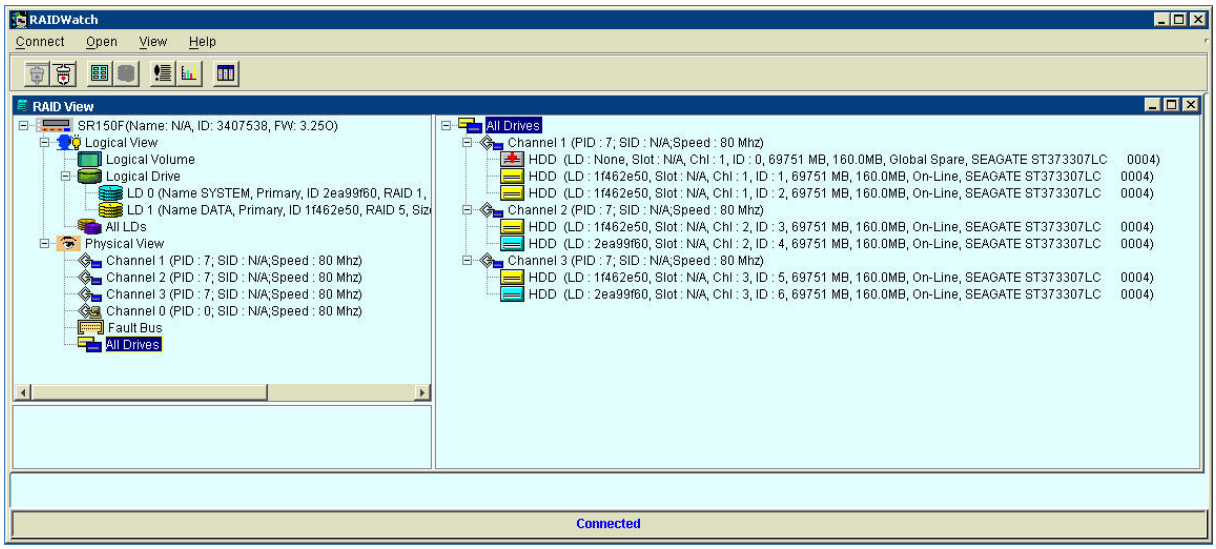


Abbildung 16 - Raid-Konfiguration (RaidWatch)

Eine detaillierte Konfiguration des RAIDs lässt sich aus folgender Tabelle entnehmen.

Kanal	SCSI-ID	Lun	Funktion
1	0	--	Global Spare , Standby
1	1	1	Raid 5

1	2	1	Raid 5
2	3	1	Raid 5
2	4	0	Raid 1 (System)
3	5	1	Raid 5
3	6	0	Raid 1 (System)

Tabelle 4 - RAID Konfiguration

Als weitere Sicherung vor Defekten kommt die S.M.A.R.T Funktion zum Einsatz.

Was versteht man unter S.M.A.R.T ?

S.M.A.R.T. steht für Self-Monitoring, Analysis and Reporting Technology und dient der Frühwarnung vor möglichen Hardware-Defekten (natürlich kann ein Gerät aber trotz S.M.A.R.T. ohne Vorwarnung ausfallen). Das RAID ist so konfiguriert, dass alle 30 Sekunden eine Überwachung stattfindet und bei einem zu erwartenden oder aufgetretenen Fehler die jeweilige Festplatte geklont und anschliessend ersetzt wird.

Im Software Lieferumfang des RAIDs befindet sich die Software RaidWatch und EventMonitor. Mit Hilfe dieser Software ist unter Windows möglich, das RAID zu konfigurieren, sowie der Überwachungssoftware mitzuteilen, was im Falle eines Fehlers gemacht werden soll. Hierbei ist es möglich eine E-Mail, Fax, SMS oder Pagernachricht zu versenden. Damit dies funktioniert, muss im Raid-Bios des Kontrollers In-Band SCSI aktiviert werden.

Was ist In-Band SCSI ?

Zur Überwachung, Konfiguration und Administration benötigen externe Geräte eine Möglichkeit zur Kommunikation mit dem Hostcomputer. Als eine Alternative zur Kommunikation über die serielle Schnittstelle (RS232), kann In-Band SCSI dienen. Dabei werden die Befehle, die zum oder vom Raid gesendet werden, in Standard SCSI-Befehle umgesetzt. Diese Befehle werden anschliessend vom Raid-Kontroller zum Hostrechner gesendet und empfangen und in entsprechende Aktionen

umgesetzt. Ein Vorteil dieser Methode ist dass die Geschwindigkeit der Datenübertragung ein Vielfaches gegenüber der seriellen Übertragung ist.

Um In-Band SCSI im Controller zu aktivieren, müssen im RAID-Bios unter dem Menüpunkt :

```
view and edit Configuration parameters
    → Host-side SCSI Parameters
        → Peripheral Device Type Parameters
```

folgende Einstellungen gemacht werden.

Name	*Wert*
Peripheral Device Type	Unknown Device (Type 0x1f)
Peripheral Device Qualifier	Connected
Device Supports Removable Media	Disabled
LUN Applicability	Undefined LUN- 0`s Only

Tabelle 5 - In-Band SCSI Raid Einstellungen

Da diese Software wie auch das CCM-System in Java realisiert sind, und bei der Installation der Überwachungssoftware die Java Runtime 1.2 bzw. bei CCM die Version 1.3 installiert werden, wurde die Überwachungssoftware ebenfalls auf die Version 1.3 umgestellt. Dies sollte zu keiner Beeinträchtigung der Funktionsweise führen, da laut der Sun-Homepage der Unterschied bei den beiden Versionen hauptsächlich in der Beseitigung von Fehlern sowie ein paar Erweiterungen besteht.

Dies wurde durch die Änderung der Verknüpfung wie in folgendem Teil zu sehen erreicht.

- RaidWatch

Original Verknüpfung:

```
"D:\Program Files\JavaSoft\JRE\1.2\bin\javaw.exe" -classpath
```

```
%CLASSPATH%;"cryptix32.jar;cryptix-
asn1.jar;ptls.jar;grm.jar" grm.GrmApplet
```

Modifizierte Verknüpfung:

```
"D:\ccm\config\java\jre1.3\bin\javaw.exe" -classpath
%CLASSPATH%;"cryptix32.jar;cryptix-
asn1.jar;ptls.jar;grm.jar" grm.GrmApplet
```

- EventMonitor

Original Verknüpfung:

```
"D:\Program Files\JavaSoft\JRE\1.2\bin\javaw.exe" -cp
"cryptix32.jar;cryptix-asn1.jar;ptls.jar;grm.jar;grem.jar"
grem.EvtWatch
```

Modifizierte Verknüpfung:

```
"D:\ccm\config\java\jre1.3\bin\javaw.exe" -cp
"cryptix32.jar;cryptix-asn1.jar;ptls.jar;grm.jar;grem.jar"
grem.EvtWatch
```

Die Funktionalität war wie zu erwarten, nach der Umstellung nicht beeinträchtigt und so konnte die Java Runtime Version 1.2 vom System entfernt werden.

3.1.3 Server Betriebssystem

In der ersten Phase der Integration wurde ein Windows 2003 Server Betriebssystem eingesetzt. Wie sich jedoch herausstellte (Funktionsstörungen und Hotline), ist diese Version des Betriebssystems aufgrund nicht funktionierender Programmteile / Funktionen (siehe Autodiscover) für den Einsatz nicht geeignet. Daraufhin erfolgte eine Umstellung auf Windows 2000 Server.

Da es keinen DHCP-Server in der Testumgebung gibt, wurde die Netzwerkkonfiguration des Servers statisch wie folgt vorgenommen.

```
Computername:  CCM-SERVER-01
Arbeitsgruppe: abb_wkgproutap
IP:            10.41.42.8
Subnet:        255.255.254.0
Gateway:       10.41.42.1
DNS Server:    10.41.14.100
               10.41.15.100
               10.41.80.100
Proxy Server:  138.223.70.10 : 8080

NIC 1 – HW-Adr. :   00-30-48-28-14-72
NIC 2 – HW-Adr.:    00-30-48-28-14-73
```

Aus Gründen der Betriebsystemsicherheit, wurden nach der Installation des Service Pack 4 für Windows 2000, folgende Patches für das Betriebsystem Windows 2000 Server installiert, welche diverse Sicherheitslücken im Betriebssystem schliessen.

Sicherheitspatches:

Security Patch ID	Knowledgebase ID	Beschreibung
MS03-008	Q814078	Fehler in Windows-Skriptmodul kann Codeausführung ermöglichen
MS03-011	Q816093	Schwachstelle in der Microsoft Virtual Machine
MS03-023	Q823559	Pufferüberlauf in HTML-Konverter kann Codeausführung ermöglichen
MS03-033	Q823718	Pufferüberlauf in MDAC
MS03-034	KB824105	Fehler in NetBIOS
MS03-039	KB824146	RPC-Schwachstelle
MS03-040	KB828026	Korrektur fuer den MS Media-Player

MS03-040	Q828750	Cumulative Patch for Internet Explorer
MS03-041	KB823182	Fehler in Authenticode Verification erlaubt das Ausfuehren von Programmen
MS03-042	KB826232	Pufferüberlauf im Windows Troubleshooter ActiveX Control erlaubt das Ausfuehren von Programmen
MS03-043	KB828035	Buffer Overflow im Windows-Nachrichtendienst
MS03-044	KB825119	Pufferüberlauf im Windows Help und Support Center
MS03-045	KB824141	Buffer Overflow ListBox und ComboBox Control

Tabelle 6 - Microsoft Windows 2000 Server Security Patches [10]

Funktionspatches:

Diese Patches beheben diverse Funktionsprobleme des Betriebssystems.

Knowledgebase ID	Beschreibung
KB820888	Computer reagiert nicht mehr (hängt) beim Versuch einen NTFS-Datenträger einzurichten, nachdem er neu gestartet wurde
KB829558	Jet 4.0 Service Pack 8
KB822831	Treiberinstallationsprogramme funktionieren nicht
Stammzertifikatsupdate	Behebt Fehler in der Stammzertifizierungsstelle
Intel Network Software Update	Behebt diverse Fehler mit Intel Netzwerkkarten

Tabelle 7 - Microsoft Windows 2000 Server Funktionspatches [10]

Die Installation der CCM Software erfolgte komplett auf die 2. Partition. Auf dem Server wurde als einziges Paket aus der CCM Reihe der Server installiert. Hierfür wurden auch gleich folgende Updates eingespielt:

- Autodiscover SR1
- CCM v.5.4.1 Services Patch 2
- Pre-OS Locator Service v 1.0

Zusätzlich wurden noch für die Betrachtung von Dokumenten und zu Sicherungszwecken folgende Programme eingerichtet:

McAffee 7.0	Antiviren Schutzprogramm
pcAnywhere 10.5	Fernsteuerungsprogramm
Norton Ghost 2003	Sicherungsprogramm
Acrobat Reader 5.0.5	Programm zum Lesen von PDF-Dokumenten
MS Office 2000 Pro	Officeprogramm (Textverarbeitung, Tabellenkalkulation, Datenbank, Präsentation)
Ahead Nero 6	CD/DVD-Brennprogramm
Winzip 8.1	Komprimierungsprogramm

Somit war die Softwareinstallation des Servers abgeschlossen.

3.1.3 CCM System Aufbau

Im nächsten Schritt musste die Umgebung für das CCM-System aufgebaut werden. Eine typische Umgebung für das System zeigt die folgende Abbildung [14].

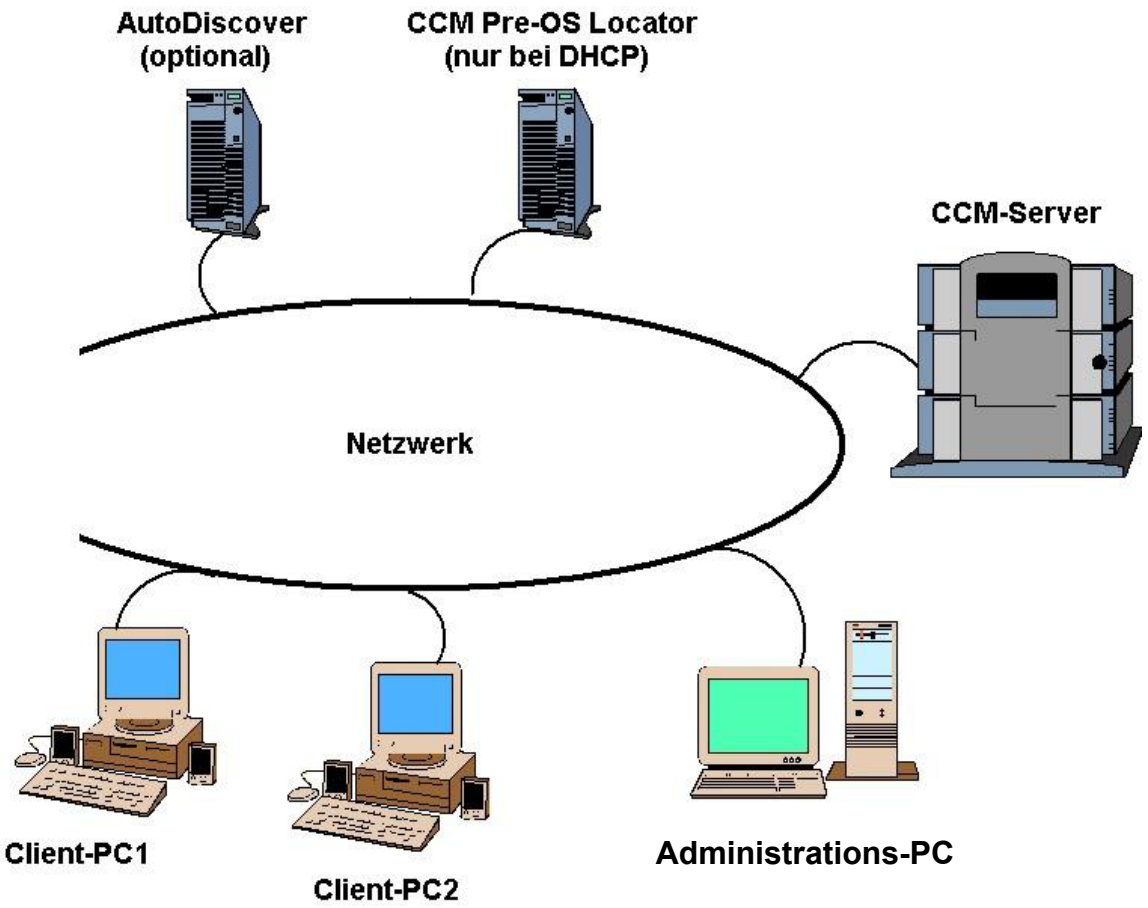


Abbildung 17 - Typische CCM-Umgebung

Installations- und Deinstallationsaufträge für die Clients werden auf dem Administrations-PC zusammengestellt. Dazu muss auf diesem das CCM Command Center installiert sein. Die zusammengestellten Aufträge werden dann in der Datenbank des CCM-Servers gespeichert.

Wurden einem Client neue Aufträge zugewiesen, so wird der CCM-Agent (auf dem Client vorhanden) vom CCM-Server über anstehende Aufträge benachrichtigt. Dieser Agent arbeitet dann die Aufträge in der Reihenfolge der Abhängigkeiten ab und meldet dem CCM-Server die Installationszustände zurück. Dem CCM-Administrator wird der Zustand der Installation dabei an seinem Administrations-PC (im Command Center) angezeigt.

Die Services PreOS-Locator (nur bei DHCP Umgebung) und AutoDiscover sind optionale Dienste, die sowohl als Stand-alone (wie in der Abbildung gezeigt) oder auch direkt auf dem CCM-Server installiert werden können.

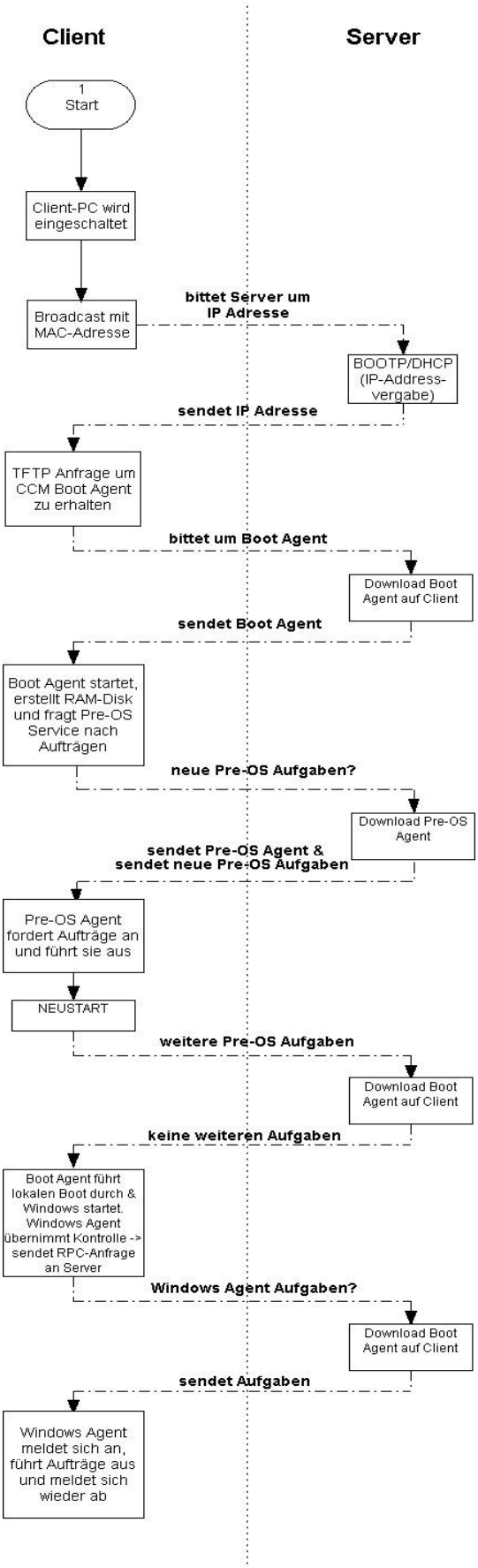
Der PreOS-Service muss dann installiert werden, wenn DHCP-Clients von verschiedenen CCM-Servern verwaltet werden sollen.
Der AutoDiscover-Service wird dann benötigt, wenn eine automatische Übernahme von Clients in das CCM-System gewünscht wird.

Client-Vorrausetzungen:

Vorraussetzung für die Verwaltung der Clients ist, dass diese mit einer Netzwerkkarte ausgestattet sind, die von den jeweiligen Betriebssystem-Skripten unterstützt wird. Diese Netzwerkkarten müssen ein CSD-, LanWorks- oder PXE-Boot-Prom ausgestattet sein. Besitzen die vorhandenen Netzwerkkarten keines dieser Boot-Proms, so besteht die Möglichkeit für diese eine Boot-Diskette mittels CCM zu erstellen. Diese Diskette ermöglicht die Kommunikation zwischen Server und Client noch bevor auf dem Client die Festplatte partitioniert und formatiert ist, beziehungsweise ein (Windows-) Betriebssystem installiert worden ist. Ansonsten gibt es keine weiteren Vorraussetzungen auf Seiten des Client-PC's. Zu beachten ist hierbei noch, dass im BIOS des Client-PC's die Bootreihenfolge (1.Bootdevice = Netzwerkkarte) geändert werden muss und die Optionen WakeUpOnLan sowie PXE aktiviert sind.

Ablauf einer CCM-Installation

Im folgenden Diagramm wird der generelle Ablauf einer Installation mittels CCM auf dem Client dargestellt [14]:



Zu Beginn sind noch keine CCM Agents aktiv. Die ersten Schritte erfolgen mittels Boot-Prom (oder Bootdiskette) und anschliessendem download des Boot-Overlay.

Hier beginnt der Einsatz des CCM Boot Agent. Er leitet die Kommunikation mit dem CCM-Server ein. Hier wird entschieden ob lokal oder vom Netzwerk gebootet werden soll. Des weiteren werden die Informationen zur Netzwerkkonfiguration eingeholt. Auf der Ramdisk wird ein DOS Mini-System platziert.

Ab hier kommt der CCM Pre-Agent zum Einsatz. Dieser Agent steuert die Aufgaben wie Partitionierung, Formatierung, Dateitransfer, starten des Setupprozess und Konfiguration der Informationsdateien für das Betriebssystem.

Ab hier übernimmt der CCM Windows-Agent die Kontrolle. Der Windows-Agent steuert die Ausführung der Softwareinstallation-, Aktions und Konfigurationspakete. Weiterhin sendet er Status- und Fehlerinformationen an den CCM-Server

3.1.4 Aufbau des Netzwerks

Nachdem der Aufbau und Installation des Servers soweit beendet war. Wurde dieser wie in der Abbildung aufgezeigt in das Netzwerk integriert. Teilweise musste das Netzwerk dabei noch durch Netzwerkverbindungen zum Switch erweitert werden.

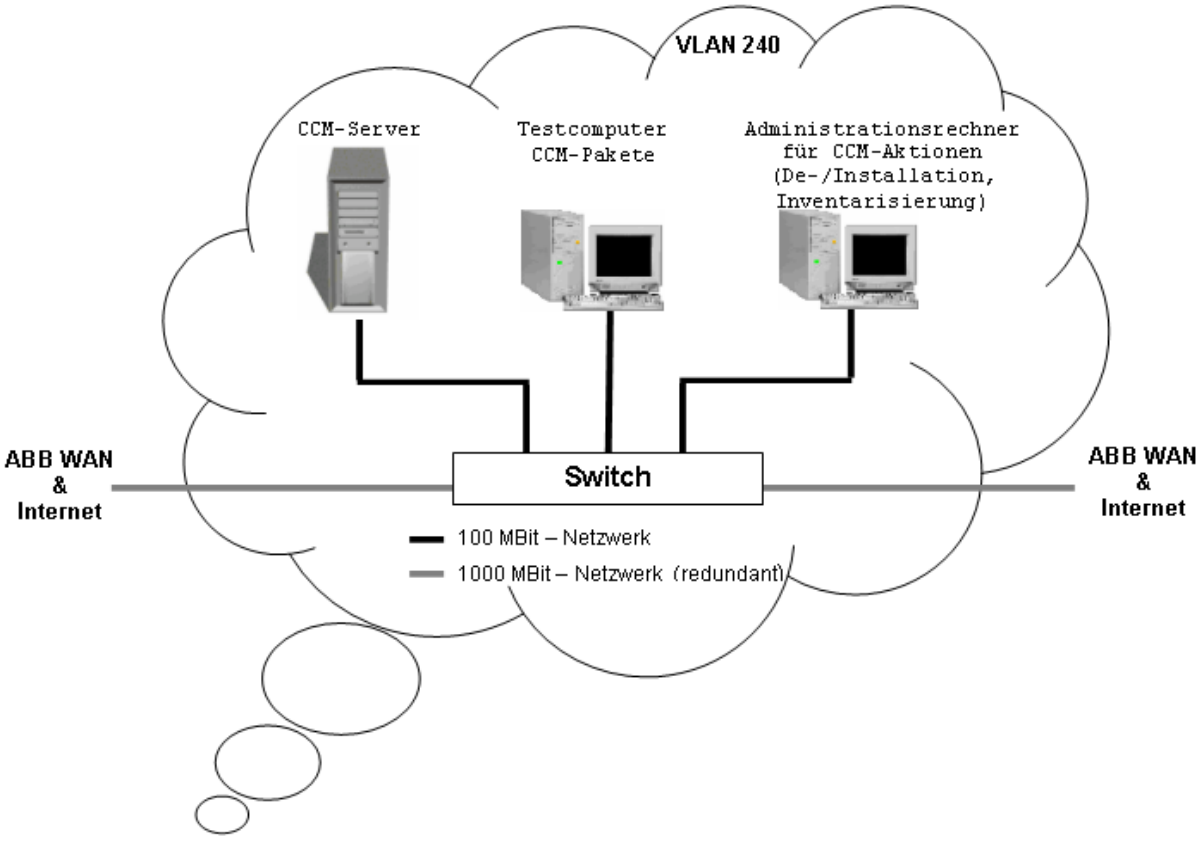


Abbildung 18 – Netzwerkstruktur

In dieser Netzwerkstruktur funktioniert das System ohne Probleme. Damit vom Command Center aus auf den CCM-Server zugegriffen werden kann, beziehungsweise dieser in der Netzwerkumgebung zu sehen ist, muss aufgrund der fehlenden Namensauflösung, in der Datei „hosts“ im Verzeichnis

```
winnt->system32->drivers->etc
```

die IP und der zugehörige Servername wie im folgenden Beispiel eingetragen werden.

Bsp: `10.41.42.8 ccm-server-01`

Eine noch ungeklärte Frage ist, wie sich das System beim Einsatz eines NAT-Routers verhält. Eventuell müssen noch weitere Ports, ausser dem CCM Standardport 484, weitergeleitet werden. Leider konnte dies nicht verifiziert werden, da der Router bis zum Abschluss der Diplomarbeit noch nicht geliefert wurde.

3.2 Skriptierung

In folgenden Kapitel wird auf die Möglichkeiten des Systems zur Erstellung der Skripte für eine vollautomatische Installation eingegangen. Die Möglichkeiten bei der Betriebssystemskriptierung sind vom jeweiligen zu installierenden Betriebssystem, beziehungsweise vom Pre-OS Agent abhäng

3.2.1 Möglichkeiten der Skripterstellung für Anwendungen (CCM)

Die in das CCM System zu integrierenden Pakete müssen zuerst skriptiert werden. Hierzu stellt das CCM-System 2 Möglichkeiten bereit. Die erste Möglichkeit besteht mittels des MSI Package Wizards, die zweite Möglichkeit besteht darin ein oder mehrere Transcripts mittels dem Development Studio zu erzeugen.

Für beide Versionen der Paketerzeugung, muss folgende Struktur auf dem Rechner welcher die Pakete in das System einbindet, vorhanden sein.

```
<PFAD>\sw\config\<paketname>        //Skriptverzeichnis
<PFAD>\depot\<paketname>            //Herstellerverzeichnis
```

Im Skriptverzeichnis werden dabei alle Dateien abgelegt die auf den Client kopiert werden müssen. Dies sind mindestens die in der folgenden Tabelle (Tabelle 8) aufgeführten Dateien, sofern diese benötigt werden. Zusätzlich können hier noch eigene Batchdateien, ausführbare Dateien usw. abgelegt werden. Im Herstellerverzeichnis werden die original Setup Dateien des Herstellers (z.B. von CD) abgelegt.

In folgender Tabelle werden die CCM Dateien, deren Funktion und wann diese erforderlich sind erklärt.

Name	*Beschreibung*	*Erforderlich für*
config.swp	Konfigurationsdatei. Enthält Informationen für die Verwaltung und Verteilung des Pakets. Die unterschiedlichen Abschnitte der Datei config.swp enthalten Daten zur Identifikation des Pakets und zu den Voraussetzungen für die Zuweisung des Pakets zu verwalteten Computern, Aufgaben, die vor oder nach der Installation oder Deinstallation der Paketdateien ausgeführt werden müssen sowie spezifische Parameter (Adminparameter z.B. für Lizenzkeys) für die Einrichtung der Paketdateien auf den verwalteten Computern.	Alle Pakete
config.dfd	Depot-Beschreibungsdatei. Indiziert, wie die Herstellerdateien kopiert werden sollen.	Alle Pakete die Software installieren
ramdisk.dsc	Nennt alle Dateien, die in die RAM-Disk des Computers geladen werden müssen.	Pakete, die für das Erstellen der RAM-Disk auf verwalteten Computern mit oder ohne Betriebssystem benötigt werden
<dateiname>.twn	Transcript/Itool Skript. Enthält alle aufgezeichneten Fenster. Daraus wird die .win Datei generiert	Anwendungs-oder Konfigurationsskript-erstellung
<dateiname>.win	Quelltext des Itool Skripts, welches die zu steuernden Dialoge enthält	Anwendungs-oder Konfigurationsskript-erstellung
<dateiname>.if	Kompilierte Version der .win Datei, welche die zu steuernden Dialoge enthält	Anwendungs-oder Konfigurationsskript-erstellung
filelist.dsc	Nennt alle Dateien, die das Paket enthält und weist jeder Datei ein Quell- und ein Zielverzeichnis zu.	Pakete, die von den Systemkomponenten installiert werden.

Tabelle 8 - Wichtige CCM Dateien

Im Folgenden Kapiteln werden die beiden Möglichkeiten der Skripterstellung genauer erklärt.

Quellen [13][14][15][16]

3.2.1.1 MSI Package Wizard

In folgenden Punkten wird auf die Erstellung und Probleme bei der Erstellung mittels des MSI Wizards eingegangen.

3.2.1.1.1 Verwendung

Wie schon in der Einleitung erwähnt kommt diese Methode nur dann zum Einsatz, wenn das Originalsetup ein MSI enthält. Für weitere Details zu MSI siehe Punkt 2.3.1. Bei Verwendung dieser Methode ist darauf zu achten, dass auf dem System auf dem später die Installation durchgeführt werden soll, auch die richtige Version des Microsoft Installers installiert ist.

3.2.1.1.2 Installation

Der MSI Package Wizard befindet sich auf einer separaten CD mit der Beschriftung MSI Package Wizard. Beim einlegen derselben startet das Setup Programm automatisch, sofern die Autostart Funktion von Windows aktiviert ist. Ist dies nicht der Fall so muss mittels des Explorers oder Arbeitsplatzes in das Wurzelverzeichnis der CD gewechselt werden und dort sie Setup.exe aufgerufen werden.
Da die Installation des Programms selbsterklärend ist, wird hierauf nicht weiter eingegangen.

3.2.1.1.3 Paketerzeugung

Nach dem Start des MSI Package Wizard öffnet sich folgendes Fenster. In diesem Fenster ist das MSI-File des zu erzeugenden oder zu modifizierenden Pakets anzugeben. Soll zusätzlich ein Transformationsfile verwendet werden, so ist dies unter dem Punkt Transformationsfile mit anzugeben.

Abbildung 19 - MSI Wizard Startbildschirm

Hat man das MSI- und eventuelle MST-File angewählt, muss zum Fortsetzen der Paketerzeugung die Schaltfläche „Next >“ angeklickt werden.

Daraufhin öffnet sich folgendes Fenster. In diesem kann man dann folgende verschiedenen Parameter für das Paket angeben

- Name: Name des Pakets der im Command Center angezeigt wird
- Package Id: PaketID im Depot (beachte 8+3 Namenskonvention)
- Classes: definiert die Klasse zu welcher das Paket gehört (damit wird ermöglicht dass andere Pakete dieses Paket benötigen, damit diese installiert werden können)
- Dependencies: definiert die Abhängigkeit von anderen Paketen (damit kann gesteuert werden welche Pakete installiert sein müssen, damit das Paket installiert werden kann)

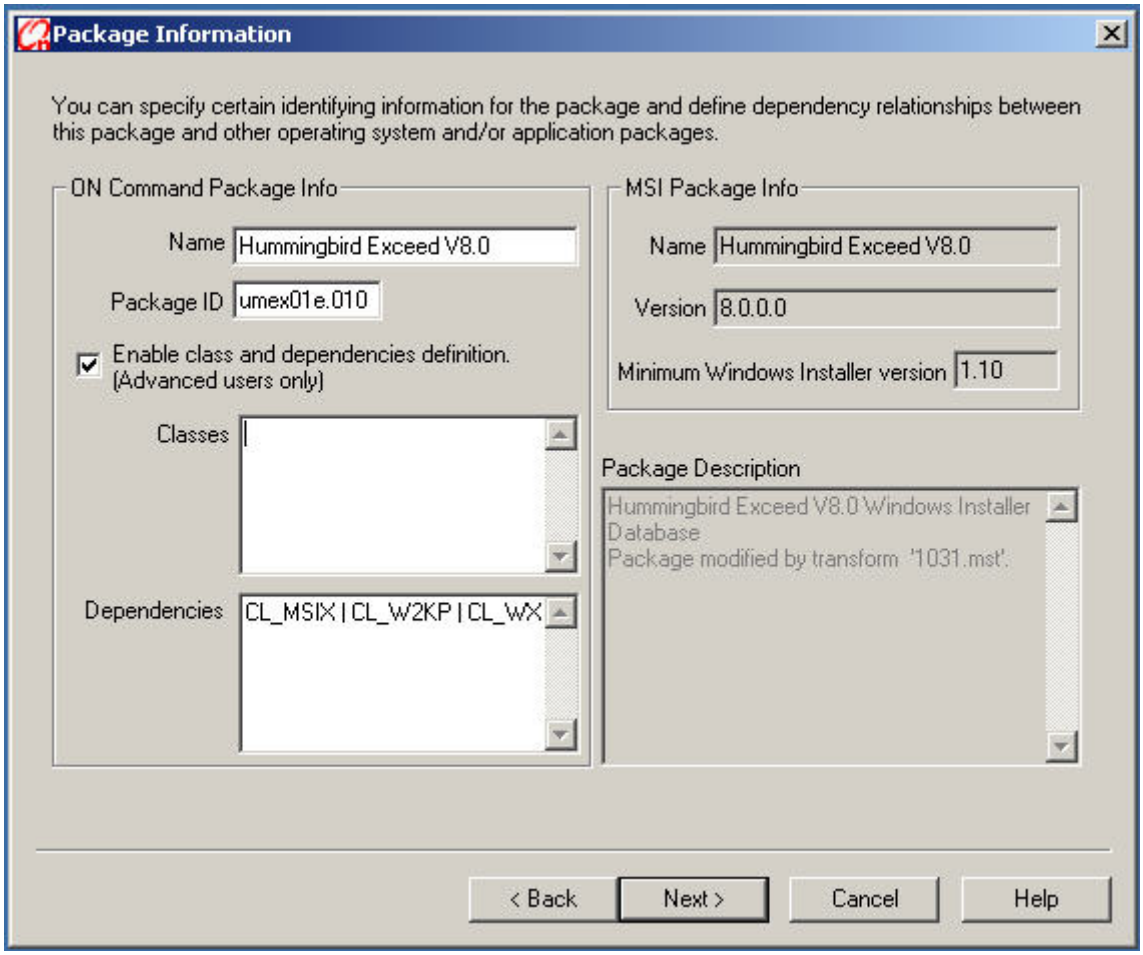


Abbildung 20 - MSI Wizard Grundeinstellungen

Hat man diese vier Grundeinstellungen gemacht, so kann man durch das Drücken des Buttons „Next“ Fortgefahren werden. Daraufhin sollte sich folgendes Fenster öffnen. In diesem Fenster sind die konfigurierbaren Properties (Eigenschaften) aufgeführt, welche durch Adminparameter zur Laufzeit konfiguriert werden können. Wählt man die jeweiligen Parameter durch das deaktivieren der Checkbox „CCM Admin Parameter“ ab, so erscheinen diese auch später in der config.swp nicht.

Abbildung 21 - MSI Wizard Standardproperties

Da diese drei Properties jedoch nicht ausreichend sind, besteht die Möglichkeit mittels des Button „Advanced…“ weitere Properties festzulegen. Klickt man auf den Button so öffnet sich folgendes Fenster.

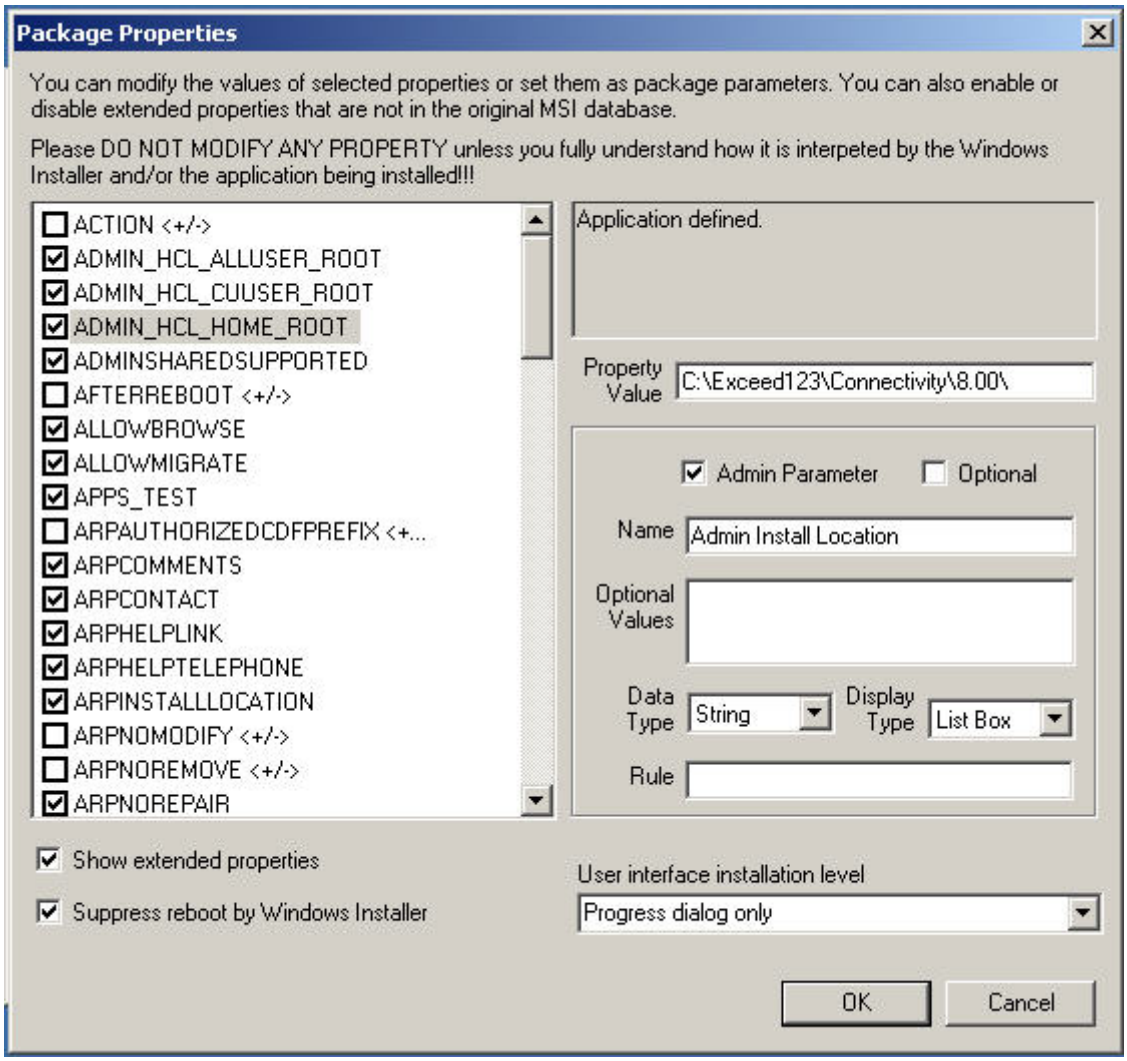


Abbildung 22 - MSI Wizard Advanced Properties

So steht z.B. INSTALLDIR für den Pfad in welches die Applikation installiert werden soll. werden. Auch besteht die Möglichkeit die für die Installation benötigten Properties als Admin-Parameter festzulegen, sodass mittels diesen die Installation später individuell angepasst werden kann. Genauere Informationen über die Properties können auf der Microsoft Homepage nachgelesen. Sind alle Properties ausgewählt so kann durch klicken auf den Button „OK“ zum vorherigen Bildschirm zurückgekehrt werden. In diesen muss anschliessend nachdem alle Properties gewählt wurden, durch klicken auf die Schaltfläche „Next“ mit der Paketerzeugung Fortgefahren werden.

In dem folgendem Dialog können nun die Installationsoptionen festgelegt werden. Dazu muss man mit der linken Maustaste auf das jeweilige Feature geklickt werden. Daraufhin öffnet sich das mittlere Fenster (siehe Abbildung 23). Soll ein Feature immer installiert werden, so der Punkt „Install locally on target computer“ ausgewählt werden. Soll es nicht installiert werden entsprechend „Do not install“. Diese beiden Möglichkeiten sind nach Abschluss der Paketgenerierung jedoch nicht mehr zu beeinflussen, d.h. wurde „Do not install“ ausgewählt, so wird dieses Feature niemals installiert.

Um dies bei der Zuweisung des Pakets zu entscheiden, gibt es den Punkt „CCM admin parameter“ (siehe Abbildung 23 mittleres Fenster). Diesem kann dann durch die zwei Optionen (siehe Abbildung 23 rechtes Fenster) ein Defaultwert zugewiesen werden, der jedoch im Gegensatz zu vorhin, bei der Installation im Command Center mittels Adminparameter nach belieben verändert werden kann (install/not install).

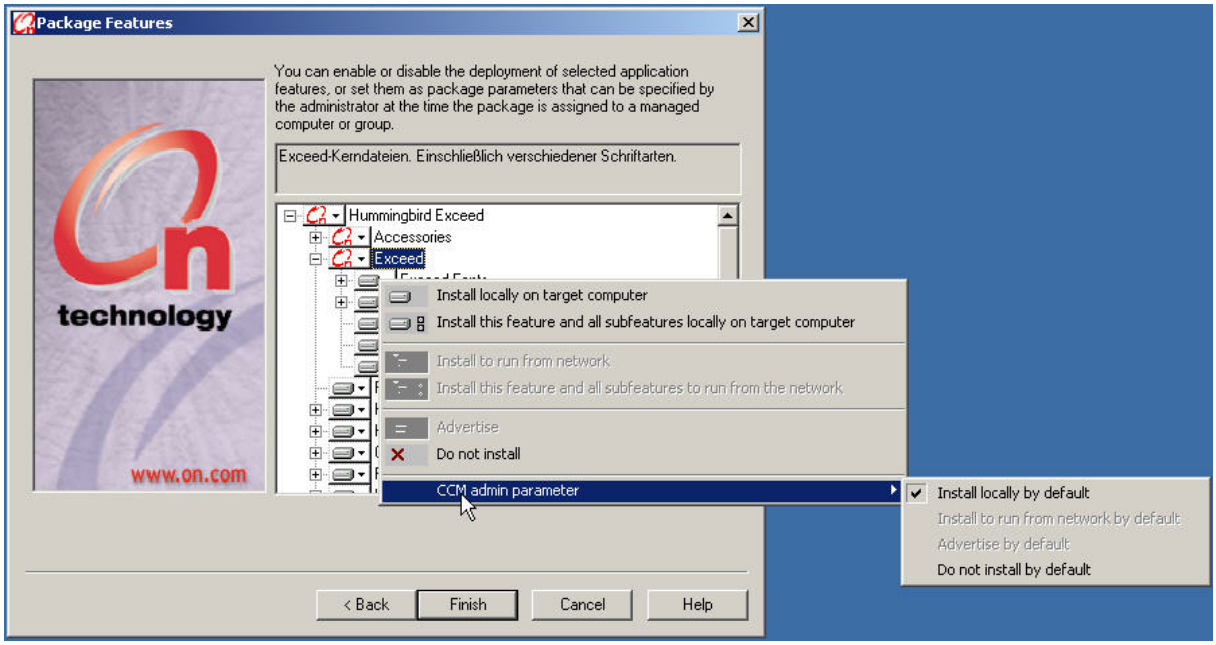


Abbildung 23 - MSI Wizard Installationsoptionen

Sind alle Installationsoptionen ausgewählt, so wird durch anklicken der Schaltfläche „Finish“ die Paketerzeugung beendet. Daraufhin sollte folgende Meldung erscheinen, ansonsten ist irgendein Fehler aufgetreten.

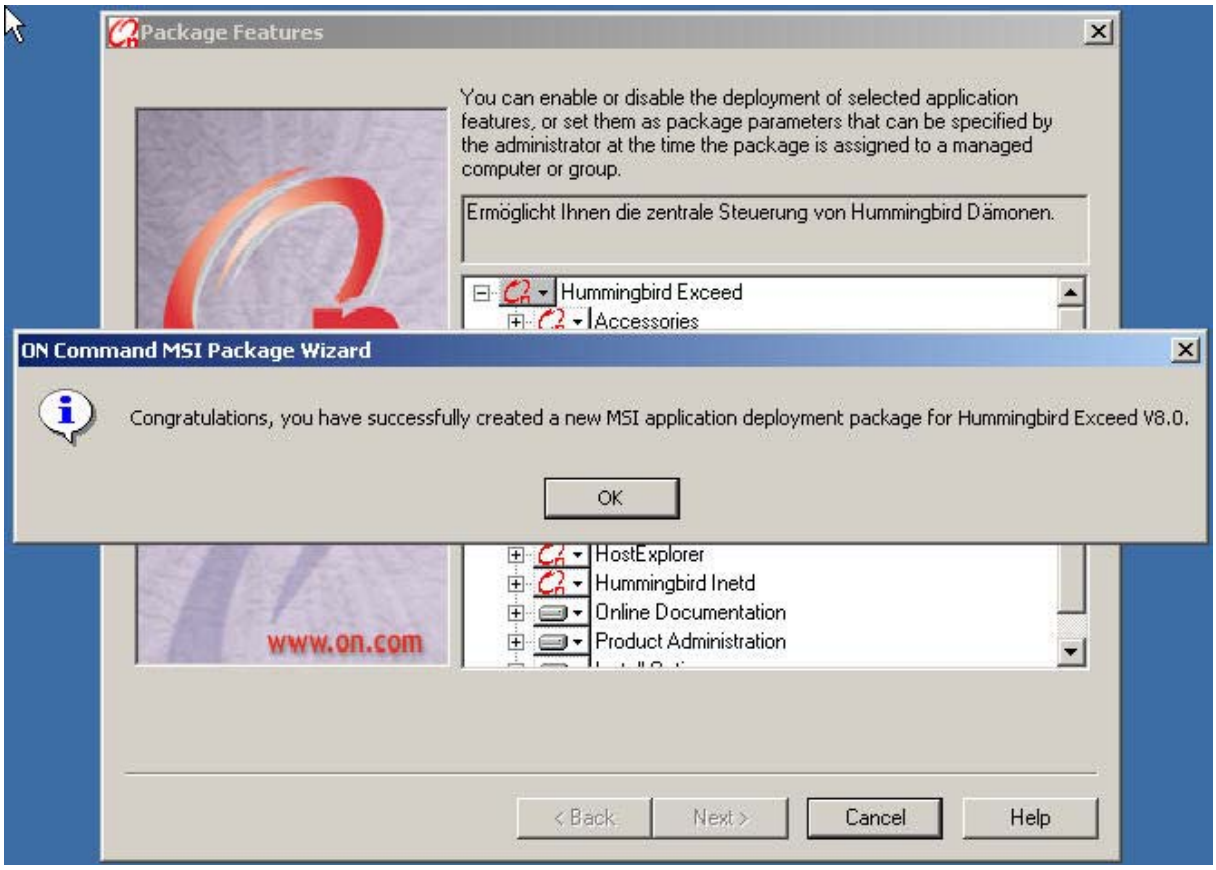


Abbildung 24 - MSI Wizard Paketerzeugung beendet

Zum Abschluss der Generierung müssen die Dateien noch kopiert werden. Auch dies erledigt der Package Wizard. Dies ist unbedingt erforderlich, da ansonsten die Konfigurationsdateien (config.swp) eventuell nicht gespeichert werden und somit das Paket auch nicht vorhanden ist. Der Benutzer wird wie in folgender Abbildung zu sehen, dazu aufgefordert.

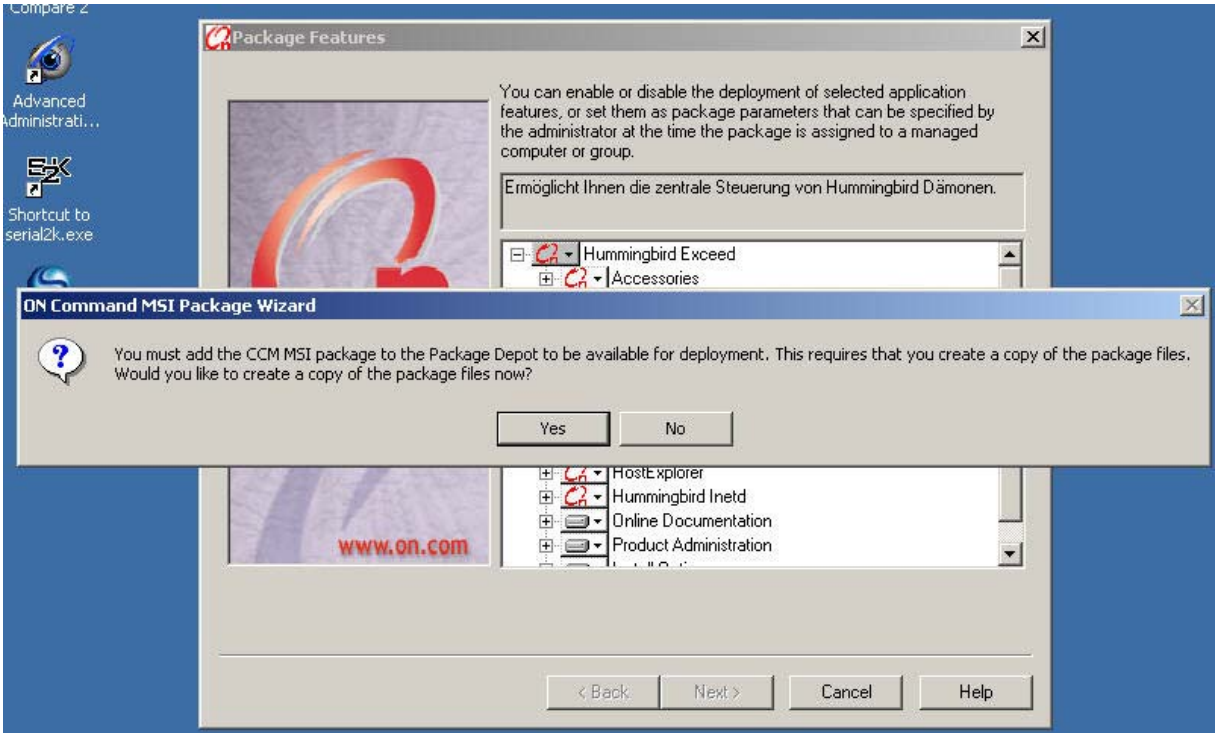


Abbildung 25 - MSI Wizard Paket kopieren

In diesem Fenster muss die Schaltfläche „Yes“ gewählt werden, woraufhin folgendes Fenster erscheinen muss, in welchem als erstes der Pfad zu den Originaldateien („Original vendor files“) angegeben werden muss.

Nun müssen noch die Pfade für die Konfigurationsfiles („Configuration Files (sw)“) sowie Depotfiles („Vendor files (depot)“) angegeben werden. Siehe auch Punkt 3.2.1 Verzeichnisstruktur

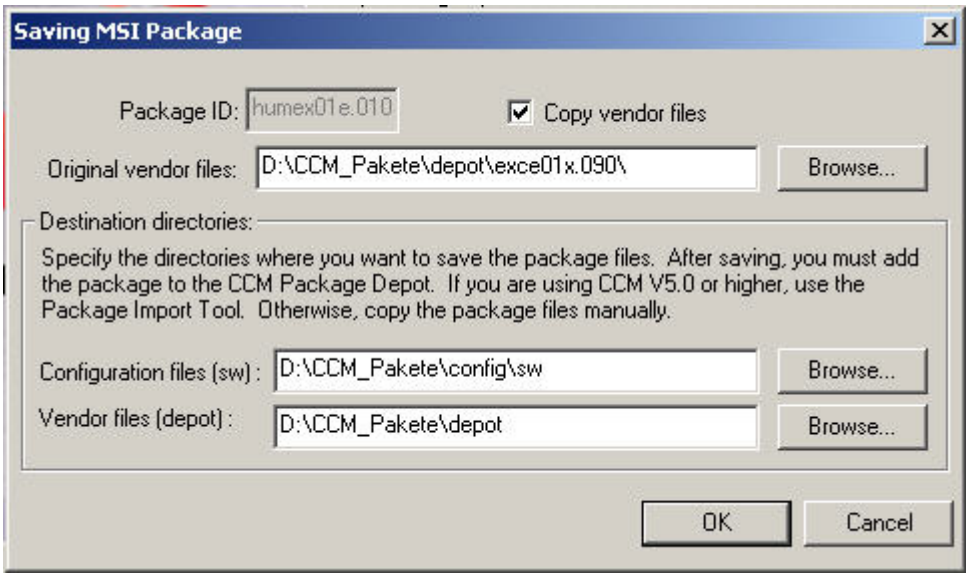


Abbildung 26 - MSI Wizard Paket speichern

Nach der Bestätigung mittels der Schaltfläche „OK“, sollte folgendes Fenster erscheinen, welches die erfolgreiche Erzeugung des Pakets bestätigt. Somit ist das Paket fertig gestellt und kann getestet werden.

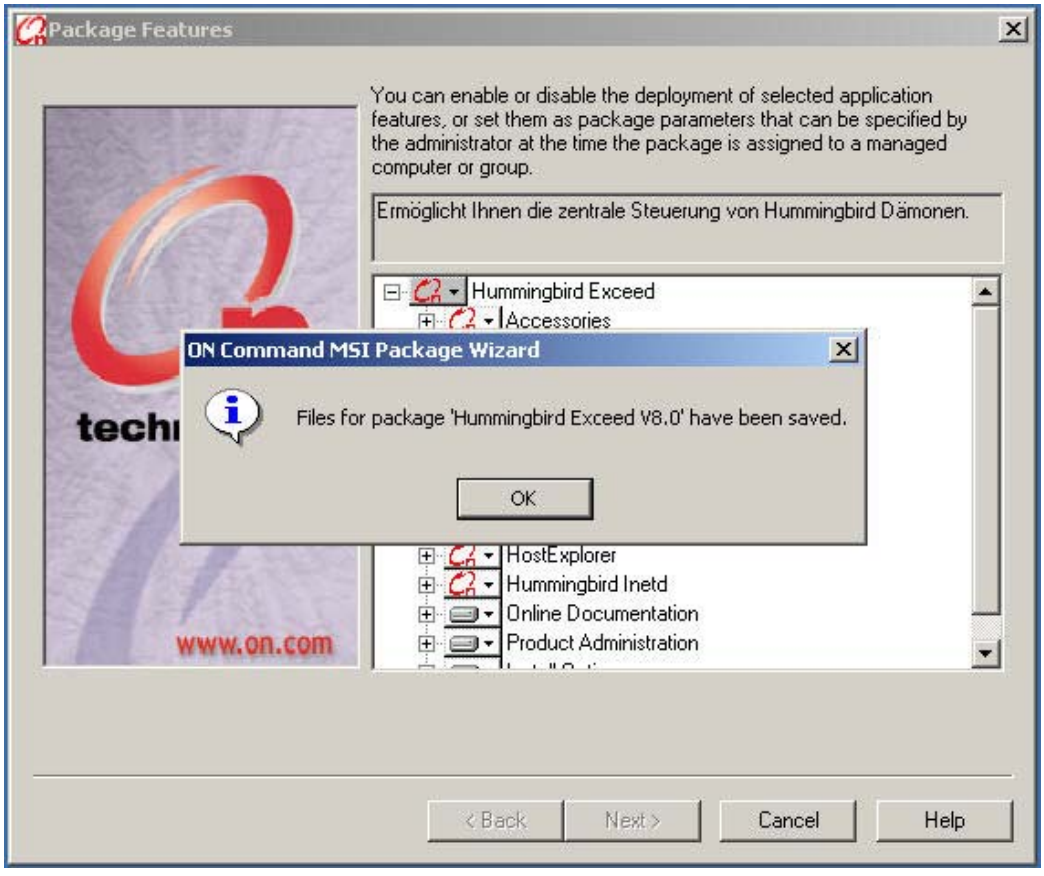


Abbildung 27 - MSI Wizard Paket fertig

Im Anschluss an die Paketerzeugung mit dem MSI Wizard sollten die im folgenden Punkt 3.2.1.1.5 aufgeführten Verbesserung in der config.swp Datei gemacht werden.

3.2.1.1.5 Verbesserungen in Config.swp

In folgenden aufgeführten Teilen der config.swp sind (eventuell) Verbesserungen vorzunehmen.

- [Info]
  Hier muss im Parameter “Version” die Herstellerversion des Pakets eingetragen werden. Standardmässig wird hier die Versionsnummer „1.0“ eingetragen

- [Admin-Param]
  Die Auswahlmöglichkeiten “Install to run from network” und “Advertise” müssen entfernt werden.

- [Post-Remove]
  Alle hier vorhandenen Einträge, welche vom MSI Wizard gemacht wurden, sollten in die [Pre-Remove]-Sektion verschoben werden

3.2.1.1.6 Testen

Die einzige Möglichkeit ein Paket, welches mit dem MSI-Wizard generiert wurde, zu testen besteht darin, dies zuerst mittels dem Package Import Tool (siehe Punkt 3.2.3) in das System zu integrieren. Daraufhin kann das Paket einem Client zugewiesen werden, und somit die Funktionalität überprüft werden.

3.2.1.1.7 Probleme

Bei den MSI Files muss es sich um „richtige“ Microsoft konforme MSI Files handeln. Dies kann aber nicht festgestellt werden. So kann es gut möglich sein dass ein scheinbar korrektes MSI File skriptiert werden kann, jedoch beim Testen die Installation nicht klappt bzw. die Parameter nicht korrekt gesetzt werden. Sollte dies der Fall sein, so kann ein Blick in die Log-Files der Installation eventuell helfen. Diese befinden sich auf dem Client, auf welchem die Installation durchgeführt werden sollte, unter dem Verzeichnis C:\integra\sw\<paketid>. Im Command Center werden meist nicht alle Informationen angezeigt, sodass es nahezu unmöglich ist beim Auftreten eines Fehlers ohne die Logfiles anzusehen den Fehler zu lokalisieren. Beim Auftreten eines Fehlers kann auch probiert werden ob der manuelle Aufruf, so wie er in der config.swp erfolgt, korrekt ist. Dazu wird am einfachsten die Kommandozeile geöffnet und der Aufruf eingefügt.

3.2.1.2 Development Studio

In folgenden Punkten wird auf die Erstellung und Probleme bei der Erstellung mittels des Development Studios eingegangen. *Quellen [13][14][15][16]*

3.2.1.2.1 Verwendung

Das Development Studio kommt dann zum Einsatz, wenn die Installationsdateien kein MSI-File enthalten, oder Probleme wie unter 3.2.1.1.7 beschrieben auftauchen.

3.2.1.2.2 Installation

Das Package Development Studio befindet sich auf einer seperaten CD mit der Beschriftung CCM Package Development Studio V1.5. Beim einlegen derselben startet das Setup Programm automatisch, sofern die Autostart Funktion von Windows aktiviert ist. Ist dies nicht der Fall so muss mittels des Explorers oder Arbeitsplatzes in das Wurzelverzeichnis der CD gewechselt werden und dort sie Setup.exe aufgerufen werden.
Da die Installation des Programms selbsterklärend ist, wird hierauf nicht weiter eingegangen.

3.2.1.2.3 Paketerzeugung

Für ein komplettes Paket ist mindestens ein ITool-Skript (siehe Punkt 3.2.1) und eine config.swp (siehe Punkt 3.2.2) Datei nötig. In den folgenden Punkten wird näher auf die Erstellung dieser Dateien eingegangen.

- ***ITool Skript erzeugen***

In einem Itool Skript, werden die Fenster und Aktionen, die während der Setupphase ausgeführt werdem aufgezeichnet Die Erzeugung eines solchen Skripts, lässt sich in folgende Teilschritte einteilen

1. Transcript aufzeichnen
2. daraus .win File generieren
3. win File testen(im Package Import Tool) & event. Modifizieren
4. win file compilieren (-> if- File)
5. eine Konfigurationsdatei (config.swp) erzeugen
6. mit Package Import Tool in die Datenbank einfügen
7. Testen

Diese Schritte werden im Folgenden erläutert.

Zuerst müssen die Einstellungen für das Projekt vorgenommen werden. Dies geschieht im Development Studio in der Menüleiste *Project* unter dem Punkt *Settings*.

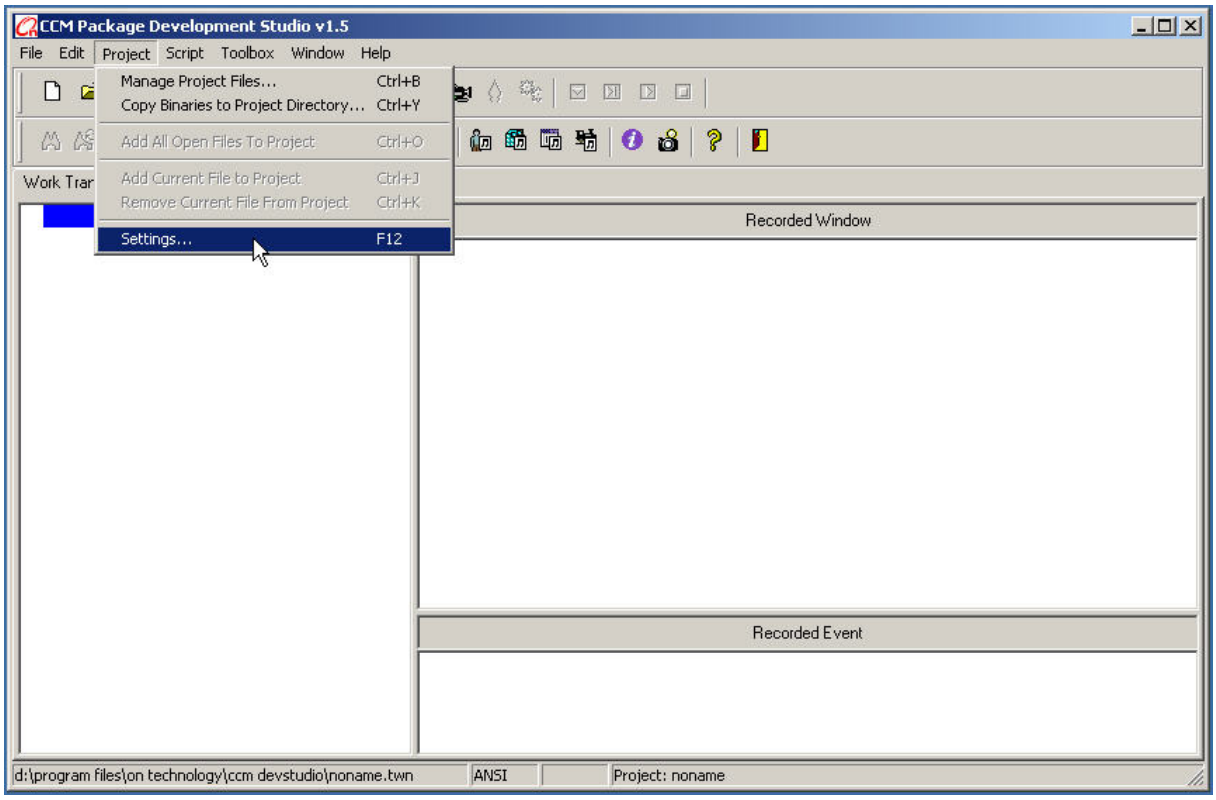


Abbildung 28 - Development Studio Erste Schritt

Daraufhin öffnet sich folgendes Fenster. In diesem ist nun das Projektverzeichnis, sowie das Depotverzeichnis anzugeben (siehe auch Punkt 3.2). Damit das Paket zu einem späteren Zeitpunkt mittels Package Import Tool in das System importiert werden kann, ist es ratsam gleich die richtigen Verzeichnisse anzugeben. Dies wären z.B. für ein Paket mit der ID test01e.010 folgende.

Project Directory d:\CCM_Packages\config\sw\test01e.010
Target Programm: d:\CCM_Packages\depot\test01e.010\1\setup.exe

Die restlichen Optionen können so belassen werden. Weitere Details können im Handbuch „Referenz für Paketentwickler“ nachgelesen werden.

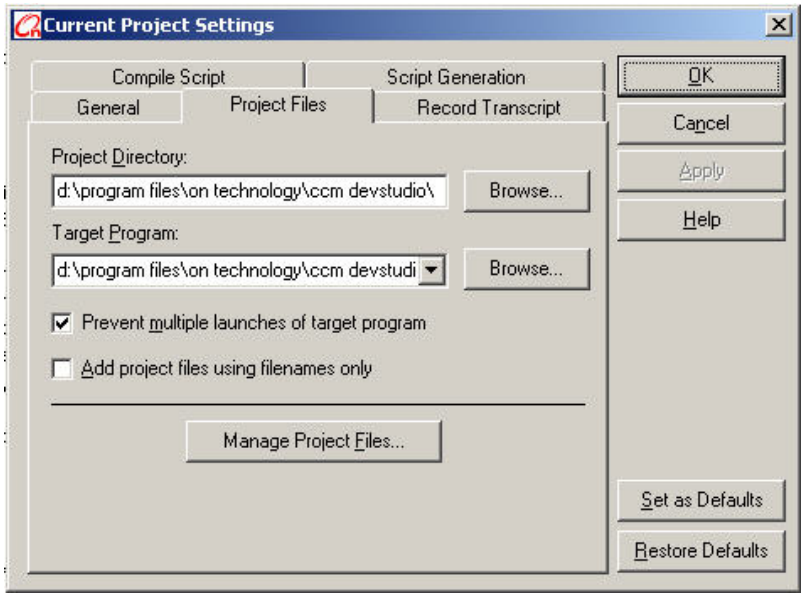


Abbildung 29 - Development Studio Projektein

Im Anschluss muss das Projekt nur noch abgespeichert werden. Der Speicherort für die Projektdatei ist für das spätere Einfügen in das System nicht relevant.

Anschliessend kann mit der eigentlichen Skripterstellung begonnen werden. Dazu wird in der Symbolleiste des Development Studio das Kamerazeichen aktiviert.

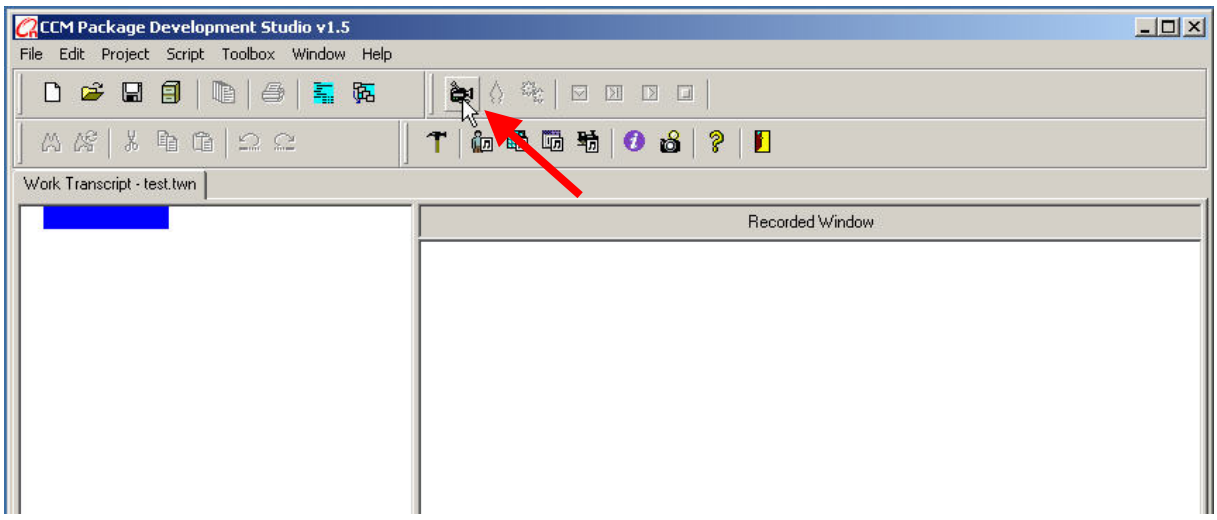


Abbildung 30 - Development Studio Aufzeichnung starten

Daraufhin sollte sich das angegebene Installationsprogramm öffnen. Am rechten oberen Bildschirmrand muss ein Fenster mit der Beschriftung CCM DevStudio

Recorder zu sehen sein. Ansonsten hat etwas nicht geklappt oder die Projekteinstellungen sind nicht in Ordnung. Ist dies der Fall, so sollte das Installationsprogramm beenden und die vorherigen beiden Schritte kontrolliert werden.

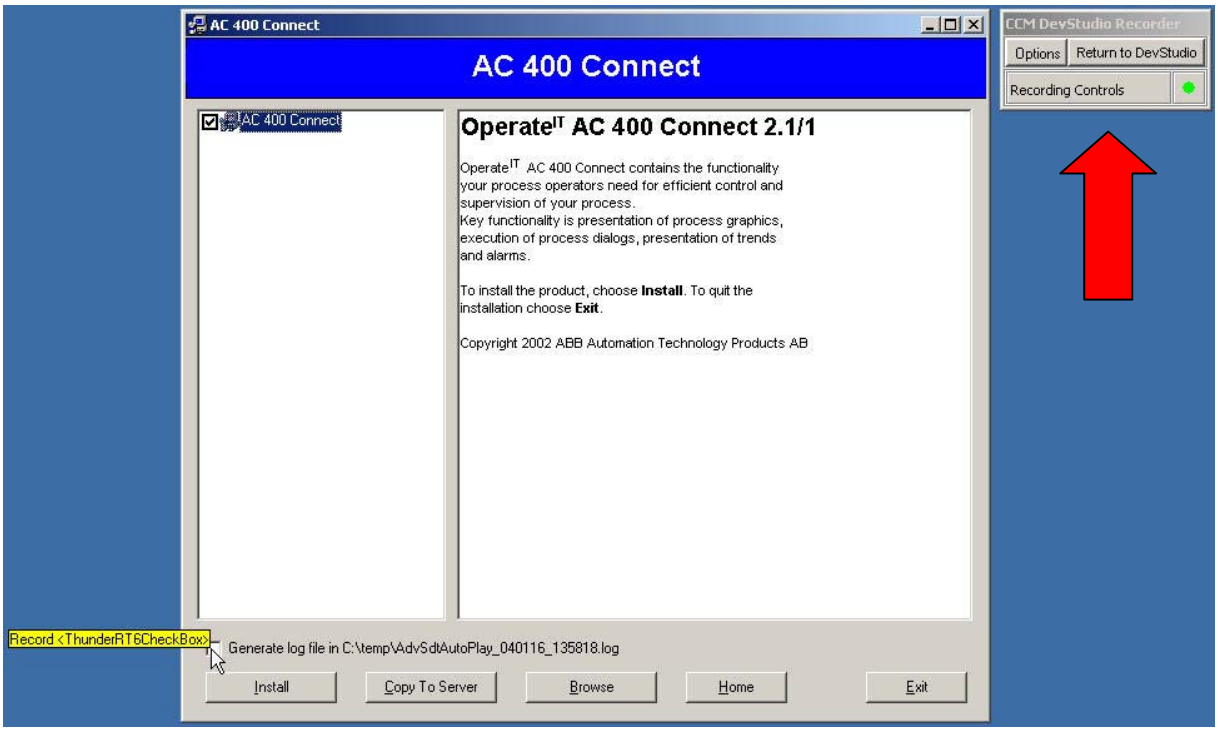


Abbildung 31 - Development Studio Aufzeichnung

In den folgenden Fenster, welche während einer normalen Installationsprozedur durchlaufen werden, muss nun jedes Eingabefeld, Kontrollkästchen und Buttonklick aufgezeichnet werden, damit dies bei der vollautomatischen Installation durch die Adminparameter gesteuert werden kann. Dies geschieht entweder durch Eingabe eines Strings bei Textfelder, oder durch klicken mit der Maus bei Optionen, Buttons und Listenfeldern.

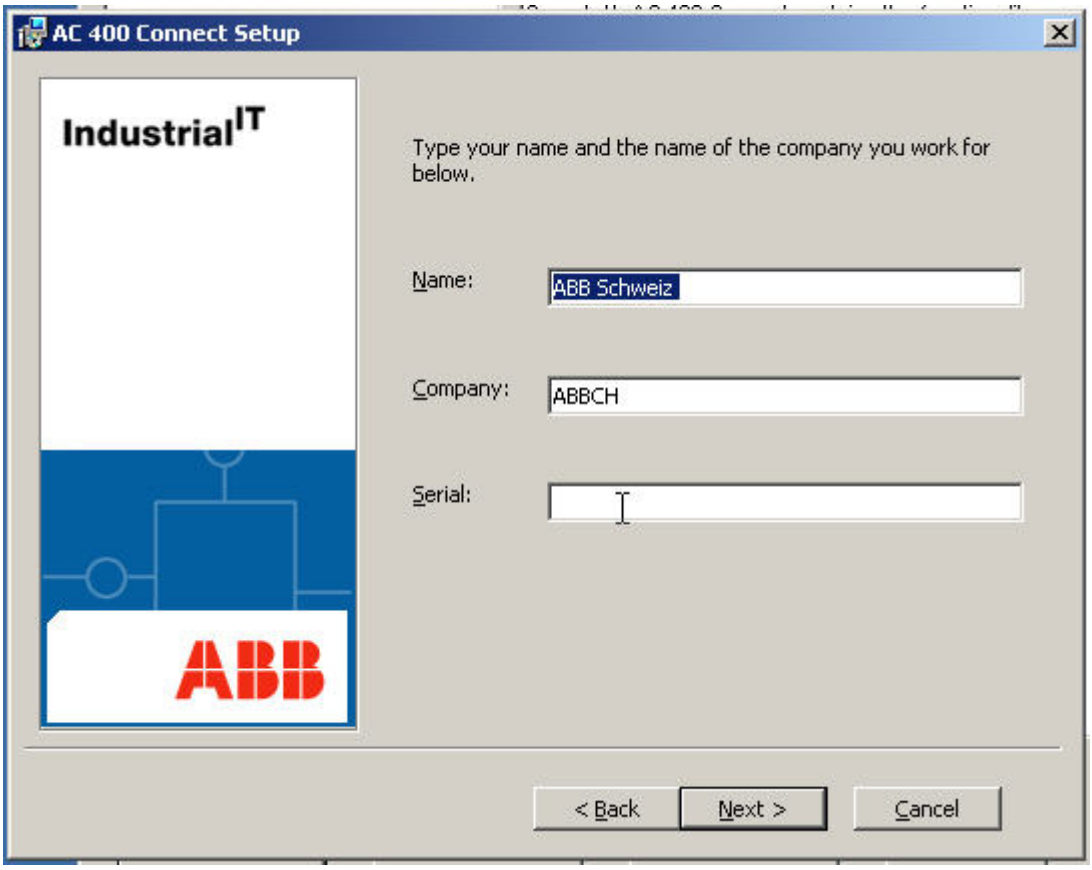


Abbildung 32 - Development Studio Beispiel Installationsprozedur

Wurde das komplette Setup durchlaufen und alles durch anklicken aufgezeichnet, so kann die Aufzeichnung durch Drücken auf den Button „Return to DevStudio“ beendet.

Abbildung 32 - Development Studio Aufzeichnung beenden

Nun sollte sich das Development Studio wieder öffnen. In diesem muss nun ein <name>.twn file vorhanden sein (siehe folgendes Bild). In diesem File sind nun alle Aktionen die man während des Setups durchgeführt hat aufgeführt.

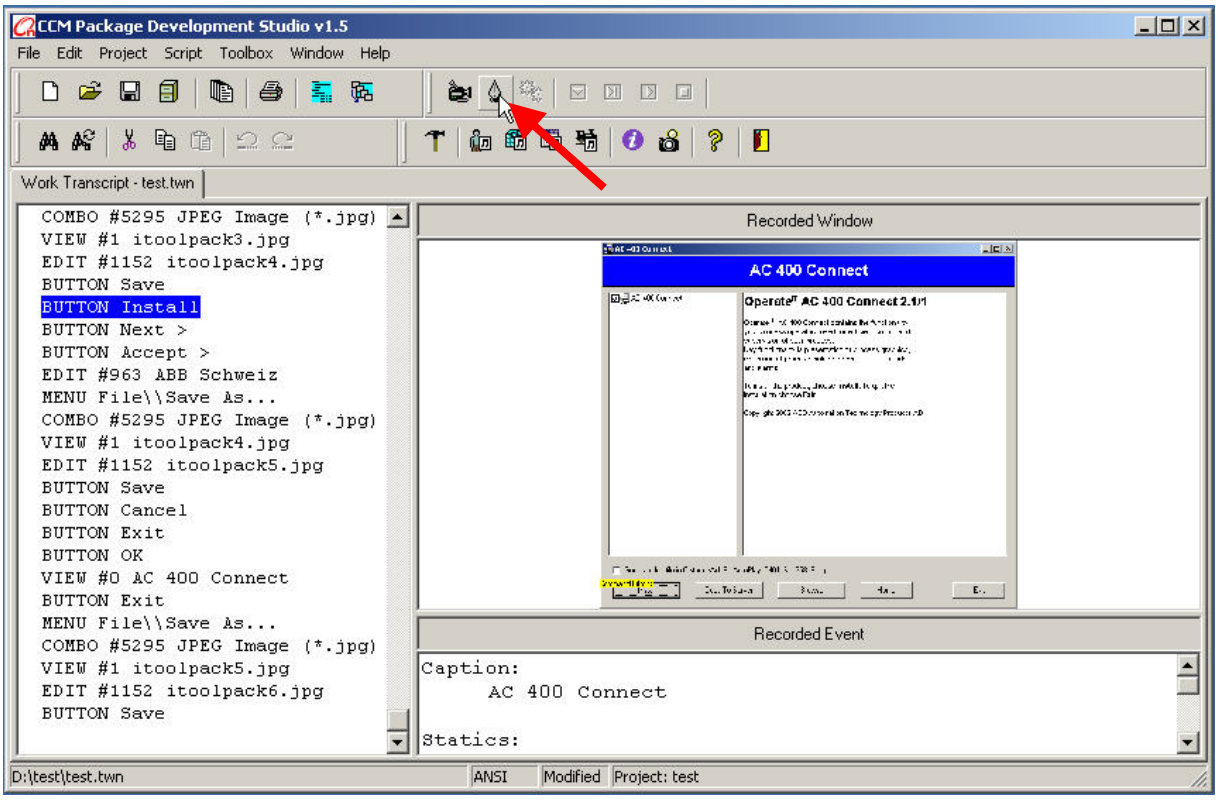


Abbildung 33 - Development Studio Transcript

Aus diesem Transcript wird nun ein win-File mittels des Buttons Feder neben der Kamera erstellt. Dieses File wird später dann kompiliert und dem Interpreter als if-File mitgegeben. Anhand dieses Files weiss der Interpreter welche Aktionen er beim angezeigten Fenster der Installationsroutine machen soll. Jetzt müssen in den aufgezeichneten Aktionen die noch statischen Eingaben durch Variablen z.B. für Pfadangaben ersetzt werden. Dies sähe z.B. folgendermassen aus:

Aufgezeichnet:

```
{„SnagIt Setup Path"}
[
     edit {"#850"} "C:\\Program Files\\SnagIT"
     button {"OK"}
]
```

Aendern in:

```
{„SnagIt Setup Path"}
[
      edit {"#850"} <PATH>
      button {"OK"}
]
```

Die Variable <PATH> wird bei der Installation des Pakets durch den zugehörigen Wert des Adminparameters (hier CCMVARSPATH) aus der config.swp ersetzt.

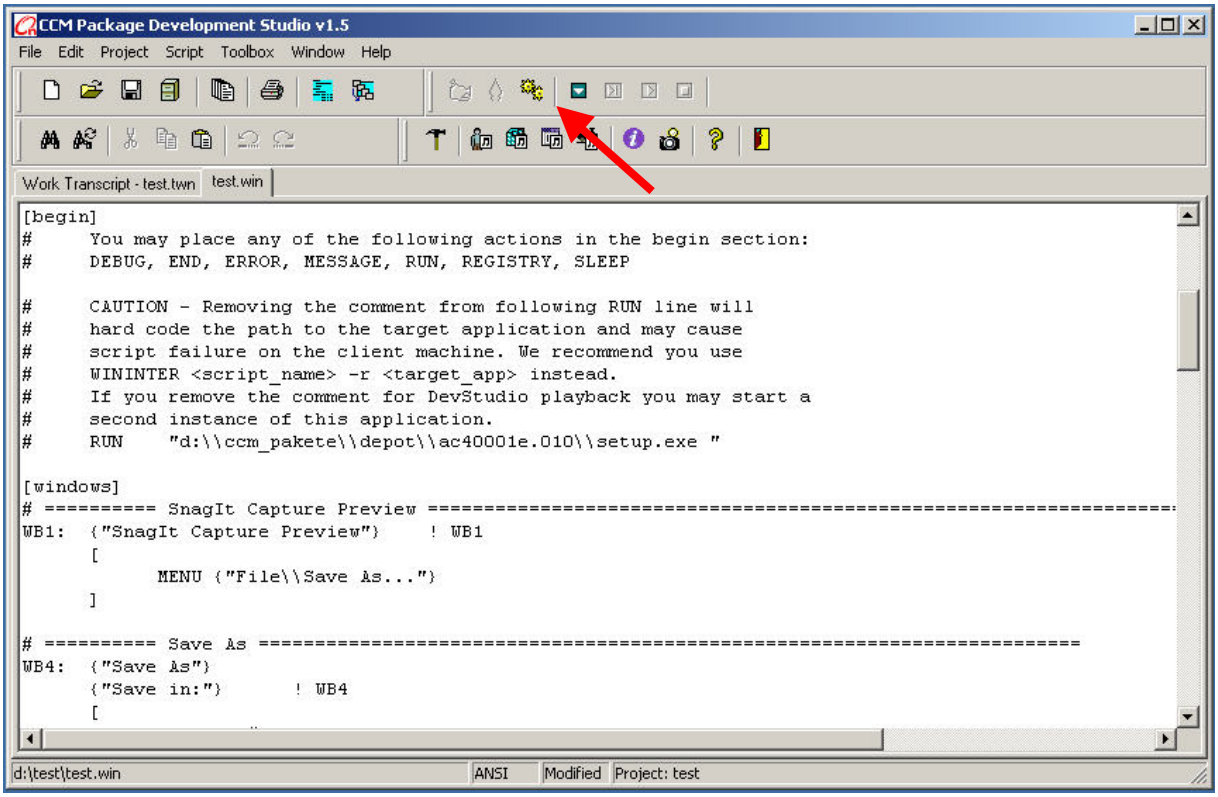


Abbildung 34 - Development Studio WIN-File

Dieses generierte File wird daraufhin durch Drücken auf den Button mit den zwei Zahnrädern kompiliert. War das win-File in Ordnung so wird nun das dazugehörige if-File erzeugt. Dies ist anhand der im folgenden Bild dargestellten Meldung zu erkennen.

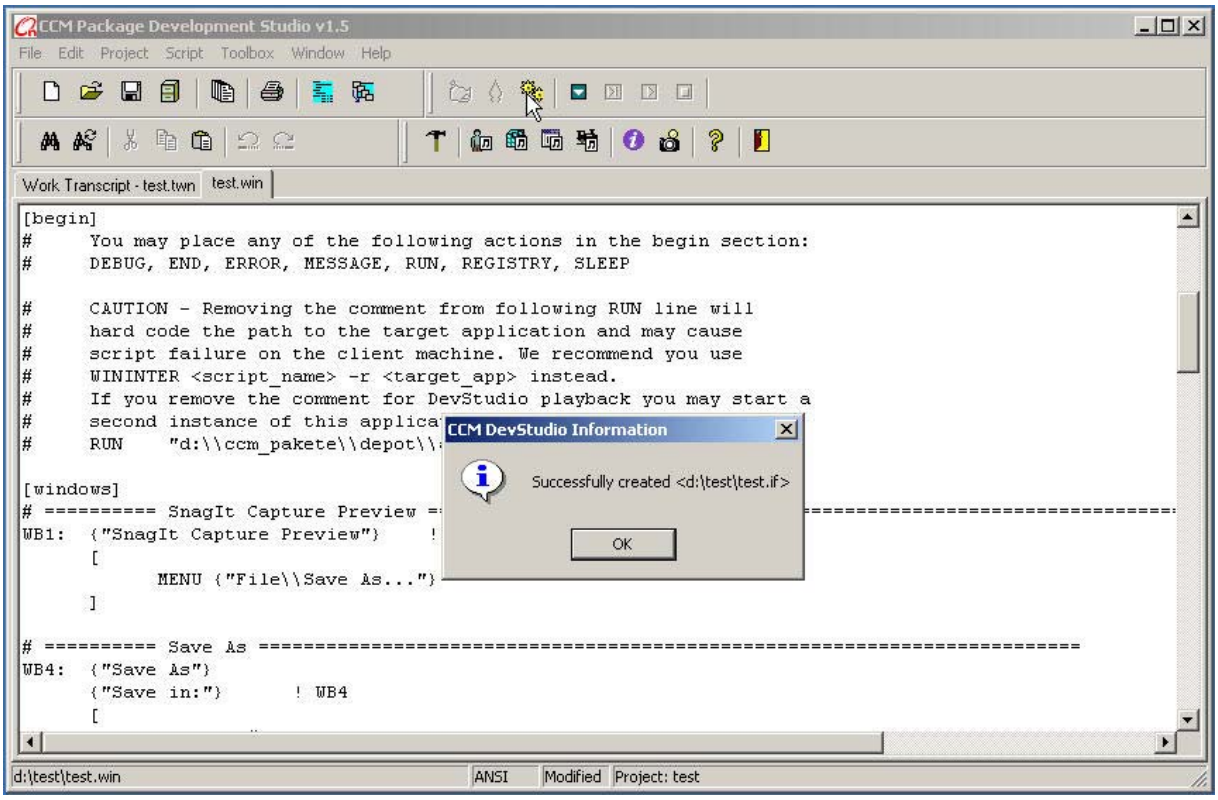


Abbildung 35 - Development Studio Skripterstellung abgeschlossen

Somit ist die Generierung des Itool Skript abgeschlossen. Anschliessend muss noch die config.swp (siehe Punkt 3.3.2 config.swp) erstellt werden.

3.2.1.2.4 Testen

Zum Testen eines ITool-Skripts kann direkt im Development Studio das Transkript zuerst kompilieren und dann ausgeführt werden. Hierzu ist folgendermassen vorzugehen. Zuerst müssen alle Schritte die in Punkt 3.2.1.2 beschrieben sind durchlaufen sein, wobei die config.swp noch fehlen darf. Konnte das ITool Skript erfolgreich kompiliert werden, so kann mittels des grünen Pfeil-Buttons neben den Zahnrädern, das Skript abgelaufen werden lassen.

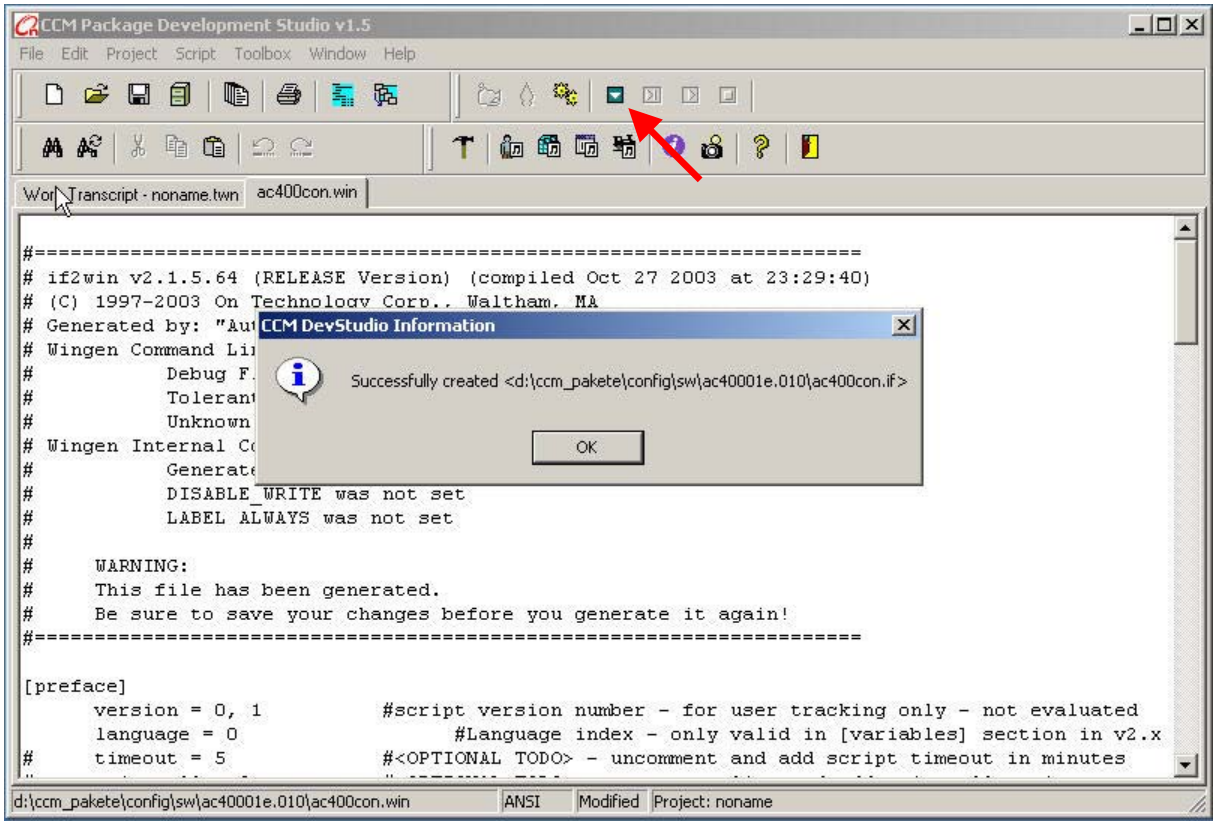


Abbildung 36 - Development Studio Skript kompilieren

Nach einmaligen klicken des Buttons erscheint folgendes Fenster. Daraufhin sollte das Installationssetup des Programms ausgeführt werden und die jeweiligen (aufgezeichneten) Aktionen ausgeführt werden. Die definierten Adminparameter (siehe Mitte rechts) müssen belegt werden, da ansonsten das Skript eventuell die Aktionen nicht ausführen kann. Unter diesem Fenster sind noch die Markierungen zur bedingten Skriptausführung aufgeführt.

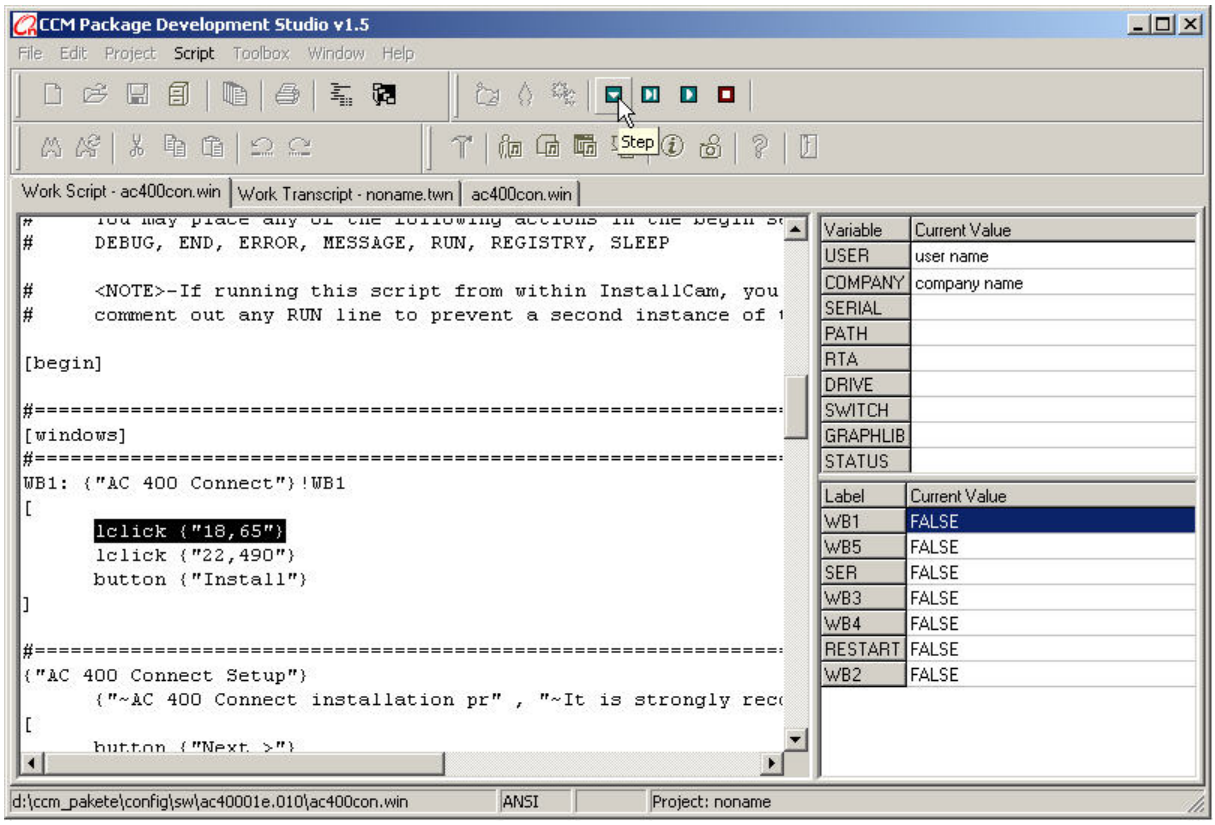


Abbildung 37 - Development Studio Skript testen

Nun muss noch entschieden werden, ob das Skript im Einzel-Step (1. grünes Icon), bis zur Cursorposition im Win-File Fenster (2. grünes Icon) oder bis zum Ende (3. grünes Icon) durchlaufen werden soll. Der rote Button dient zum Abbruch des Skripts.

War dieser Test erfolgreich so muss um das komplette Paket zu testen, dies mittels dem Package Import Tool in das System integriert werden (siehe Punkt 3.3.3). Natürlich muss zu diesem Zeitpunkt bereits die config.swp erstellt worden sein. Daraufhin kann das Paket einem Client zugewiesen werden, um zu sehen ob das Paket funktioniert. (Siehe hierzu auch Punkt 3.4 Probleme)

3.2.2 Config.swp

In der config.swp wird nun der Installationsablauf, sowie die Administrationsparameter (Adminparameter) für das Paket festgelegt. Die im Itool Skript eingefügten Variablen werden durch den Wert dieser Adminparameter ersetzt. Diese können bei der Zuweisung eines Pakets in einer komfortablen Maske gesetzt werden. Eine neue Konfigurationsdatei wird unter dem folgendem Menüpunkt erstellt:

```
File->New->Configuration Package->Configuration Package Template
```

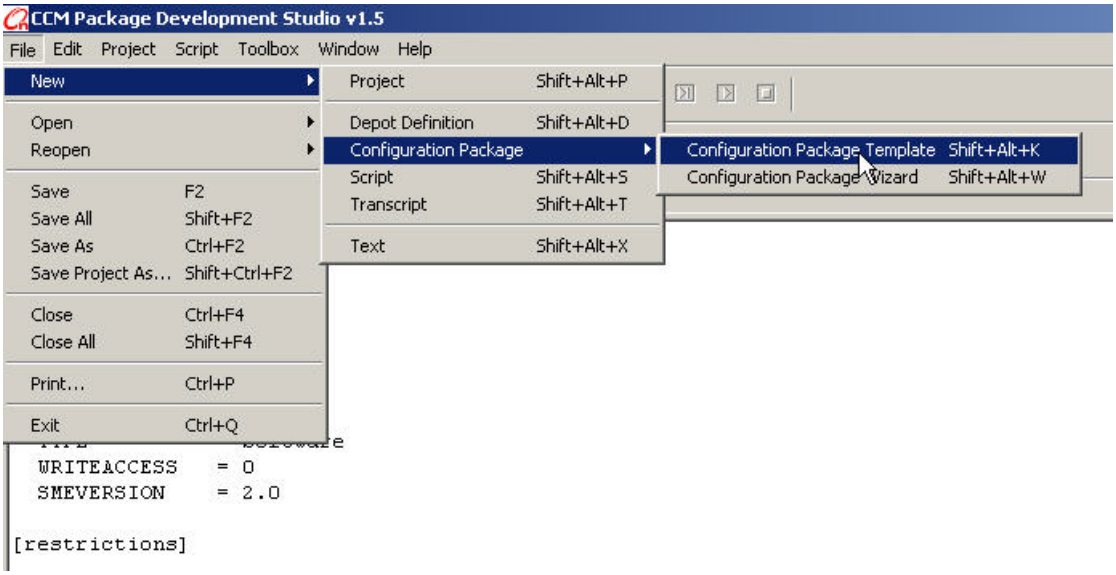


Abbildung 38 - Development Studio Config.swp erstellen

In dieser Datei muss nun im Teil [info] der Name des Pakets unter dem Punkt NAME eingetragen werden. Unter diesem Namen wird das Paket im Command Center angezeigt.

```
[info]

  NAME            = Demo test package v.1.0 (English)
  VERSION         = 1.0
```

Des Weiteren ist im Teil `[pre_install]` oder `[post_install]` das aufzurufende ITool-File anzugeben. In diesen Teilen des Skripts können auch andere Aufrufe, die

das jeweilige Betriebssystem unterstützt (z.B. Batchdateiaufruf) oder Skripte usw., erfolgen. Der Aufruf für das obige test.win Skript sähe dann folgendermassen aus:

```
[pre_install]
     wininter.exe test.if –r"\\ccm-server-
                          name\r1st_20\<paketID>\1\setup.exe"
```

Wininter.exe ist der Interpreter für das ITool Skript, test.if ist das kompilierte Skript und der letzte Parameter –r"<Pfad zum Server>\<Programmname>" beschreibt das auszuführende Programm bzw. die Setuproutine.

Im Teil `[admin param]` müssen nun noch die unter 3.3.1 erwähnten ersetzen festen Strings, die zugehörigen Variablen definiert werden. Diesen im Skript verwendeten Variablen muss in der config.swp noch der Präfix "CCMVARS" vorangestellt werden, damit das System weiss dass diese an das Skript weitergeleitet werden sollen.

Nun kann das Paket wie unter Punkt 3.2.3 beschrieben in das System integriert werden.

3.2.3 Package Import Tool

Das Package Import Tool dient zum importieren (bzw. updaten) neuer Pakete in die Server Datenbank. Das Einfügen der Pakete muss immer mittels dieses Tools erfolgen. Die Dateien werden dabei auf dem Server folgendermassen abgelegt.

```
%CCMHOME%\sw\config\<paketname>    //Skriptverzeichnis
%CCMHOME%\depot\<paketname>   //Herstellerdateien
```

Das Einfügen eines solchen Pakets erfolgt durch einfaches Drag & Drop oder Copy & Paste vom lokalen Rechner zum Server. Die Oberfläche ähnelt dabei, wie im folgenden Bild zu sehen dem Command Center.

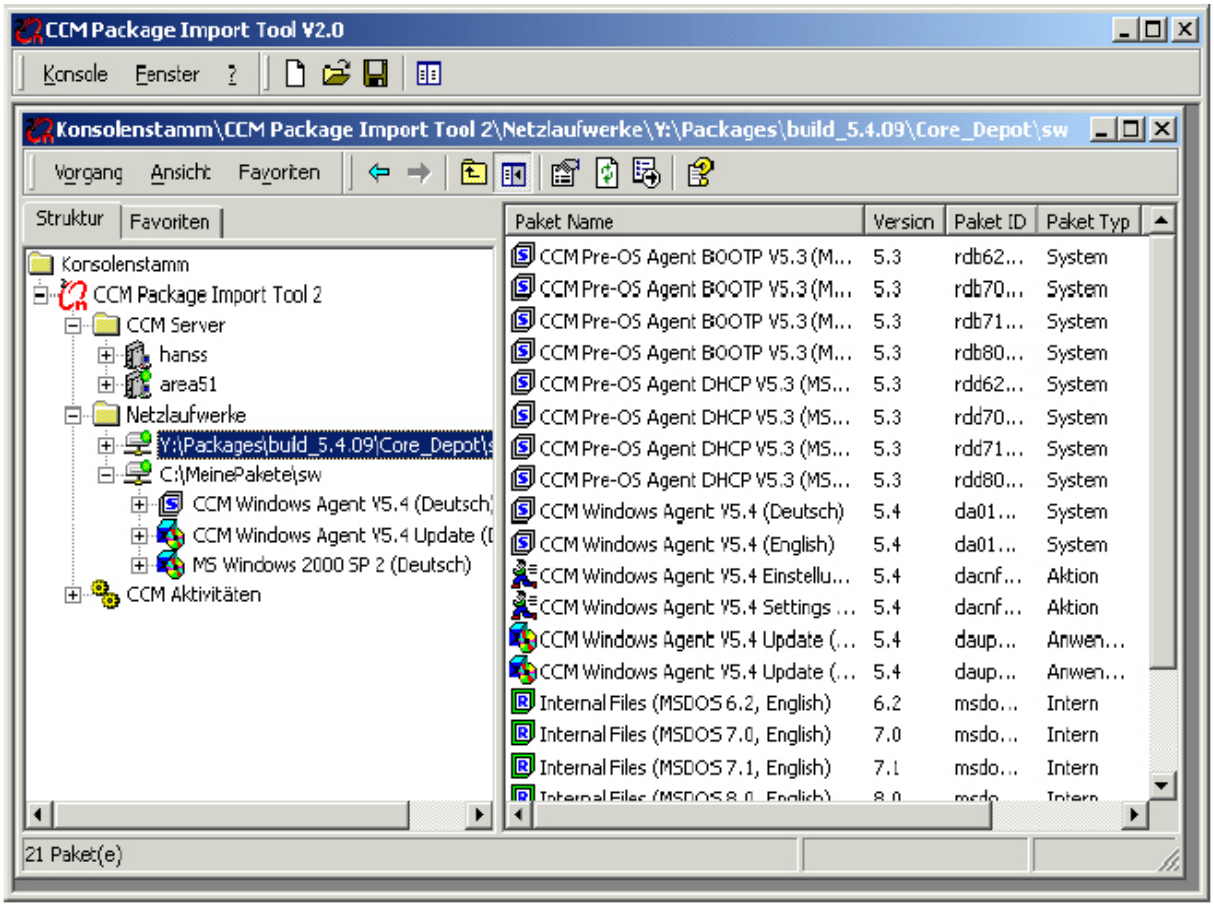


Abbildung 39 - Package Import Tool

3.3 Umsetzung

In folgenden Kapiteln werden die Erweiterungen bzw. Konzepte für die Pakete die erarbeitet wurden näher beschrieben.

3.3.1 Betriebsystempakete (Windows NT/2000/XP)

Die Firma OnTechnology stellt mit der Auslieferung der Software bereits fertige Skripte zur Verfügung. Für die Betriebssysteme Windows NT, 2000 und XP existieren bereits fertige Skripte mit denen einem Client ein Betriebssystem zugewiesen werden kann. Dabei wird wie im Kapitel 2.3.1 (Windows 2000/XP/NT) beschrieben, eine unattended Setupdatei zur Laufzeit und in Abhängigkeit der Adminparameter erstellt. Daraufhin wird das komplette Installationsverzeichnis (i386)

der Windows Version auf den Client kopiert, und anschließend das Setup lokal gestartet.

Somit mussten die Betriebssystemskripte nur noch an die eigenen Bedürfnisse angepasst werden. Aufgrund der geringen Treiberunterstützung des Windows Setups, beziehungsweise neuer Treiberversionen, sollte eine Lösung zur Integration zusätzlicher Treiber gefunden werden.

Damit dem unattended Setup neue Treiber zur Verfügung stehen, müssen folgende Vorraussetzungen erfüllt sein. Erstens muss das Setup wissen, wo sich die zusätzlichen Treiber befinden, und zweitens müssen diese Treiber auf den Rechner kopiert werden damit diese lokal vorhanden sind. Damit das Setup weiss wo sich die neuen Treiber befinden, müssen die Verzeichnisse in der unattended Setupdatei (params.txt) angegeben werden.
Bezüglich des zweiten Punkts, wäre die einfachste Lösung alle Treiber in ein Verzeichnis zu legen, welches anschliessend auf den zu installierenden Rechner kopiert wird. Angesichts der Hardwarevielfalt, und der damit verbundenen unnötigen Netzwerkbelastung aufgrund der Datenmenge, kann dies jedoch nicht die Lösung sein.
Daher, sowie aus Gründen der Übersicht, wurde für die verschiedenen Treiber im Depotverzeichnis auf dem Server zunächst einmal folgende Verzeichnisstruktur erstellt.

- `nic` für Netzwerkkarten
- `video` für Grafikkarten
- `audio` für Audiokarten
- `modem` für Modem- und ISDN-Karten
- `chipset` für Chipsatztreiber
- `other` für sonstige Treiber

(Stammverzeichnis der Treiber <pfad zum Paketdepot>\5\pnp\)

In diesen Verzeichnissen werden dann für jedes Gerät bzw. jeden Treiber ein entsprechendes Unterverzeichnis angelegt.

So z.B. für eine Matrox Grafikkarte G450/550 das Unterverzeichnis `matxg450` im Verzeichnis `<pfad zum Paketdepot>\5\pnp\video`. In dieses Verzeichnis müssen anschliessend die Treiberdateien kopiert werden. Bei der Namensvergabe für das Treiberverzeichnis ist darauf zu achten dass der Verzeichnisname aus maximal 8 Zeichen besteht, da das unattended Setup ansonsten die Treiber aufgrund der 8.3- Dos-Konvention die Dateien nicht finden kann. Der verwendete Verzeichnisname ist auch gleichzeitig der Wert der Adminvariablen, womit es daraufhin möglich ist, über diese Parameter die zu installierenden Treiber auszuwählen.
Damit nicht jeder Treiber einzeln ausgewählt werden muss, sollte es des Weiteren möglich sein, einen PC-Typ auszuwählen woraufhin diesem automatisch die richtigen Treiber zugeordnet werden. Dies wird durch eine Batchdatei erreicht, in der die entsprechenden Umgebungsvariablen (entsprechend den Adminvariablen), gesetzt werden. Für diese Batchdateien wird im Depotverzeichnis ebenfalls ein Unterverzeichnis mit der Bezeichnung `pctype` angelegt, in dem diese dann gespeichert werden. Die Einträge in einer solchen PC-Typ Datei haben folgende Struktur.

Beispiel (`<pfad zum Paketdepot>\5\pnp\pctype\t1300.bat`):

```
REM ---------- PNP-Drivers for Transtec 1300 PC ------------
set NIC1=i82550
set NIC2=none
...
set VIDEO1=matxg450
...
```

Bei der Ausführung der config.swp wird nun geprüft ob ein PC-Typ (z.B. Transtec 1300) ausgewählt wurde, woraufhin die entsprechende Batchdatei (t1300.bat) zum setzen der Variablen aufgerufen wird. Ist dies nicht der Fall werden die Adminvariablen abgefragt ob zusätzliche Treiber installiert werden sollen. Abhängig davon werden die Treiber auf den Client kopiert.

Folgendes Beispiel soll die Vorgehensweise zur Integration neuer Treiber verdeutlichen:

- Treiberinformation: neuTreib.inf
- Treiberart: test
- Treiberverzeichnis: nTreib1

1.neues Verzeichnis im Depot mit Bezeichnung nTreib1 anlegen

```
<pfad zum Paketdepot >\depot\w2pos05g.050\5\pnp\test\nTreib1
```

2.Treiber ins Depot kopieren

3.Anpassungen der config.swp

```
[admin param]

.......

#MH------------------------- Additional Driver ----------------------------
      NEU1  -n"Neue Treiber 1" -r"SEL_PCTYPE.select" -d"LIST" = <strtup> \
                               "none",     "kein neuer Treiber";\
                               "nTreib1",  "Neuer Treiber 1";
.......
```

```
[post-install]

.......
#------------------MH Copy Drivers-------------------#

### Verzeichnis anlegen
if not exist %WINDRV%\drivers\test\nTreib1 md %WINDRV%\drivers\test\nTreib1
> NUL

### Treiber kopieren
  if "%NEU1%" == "none" goto end_n1
            smecp16 -s \\depot\%W2KPDEPOT%\5\pnp\test\%NEU1%\*.*
                       %WINDRV%\drivers\test\nTreib1
  :end_n1
.......
```

4. Anpassen Params.txt

```
OemPnpDriversPath=\drivers\nic\nic1;....;\drivers\test\nTreib1;
```

Soll ein neuer PC-Typ definiert werden, so muss lediglich die Batchdatei in das Depot kopiert werden, und in der config.swp in den Adminparametern ein entsprechender Eintrag hinzugefügt werden.

Beispiel:

Hinzugefügt wird der PC-Typ IBM PC Model 0815 mit der Batchdatei ibm0815.bat

[admin param]

```
#MH------------ Drivers for special machine/pctype----------
  SEL_PCTYPE -n"Select PC Type/drivers?" -d"LIST" = <strtup> \
              "none","No additional Drivers";\
              "select","Select components seperately";\
              "tran1300","Transtec 1300 - Asus Mainboard";\
              "ibm0815","IBM PC Model 0815";
```

Die Erweiterung des Skripts, sowie ein Beispiel für die Definition eines PC-Typs kann im Anhang G – Treiberintegration eingesehen werden.

3.3.2 Anwendungspakete

Mit den unter den Punkt 3.2.1.1 MSI Wizard und 3.2.1.2 Development Studio beschriebenen Möglichkeiten zur Automatisierung der Anwendungsinstallation, wurden alle im Anhang A (Eigene Pakete) aufgeführten Anwendungen erstellt. Im Anhang finden sich zudem noch Stichpunktartig Informationen über die Besonderheiten der einzelnen Pakete.

3.4 Probleme bei der Umsetzung

In folgenden Kapiteln werden die Probleme bei der Umsetzung bzw. Skripterstellung näher beschrieben.

3.4.1 Standardprobleme

Bei der Erstellung der Anwendungspakete tauchten des Öfteren folgende Standardprobleme auf, die sich durch hängen bleiben des Skripts oder Fehlermeldung im Command Center äusserten:

- **Steuerelemente werden nicht erkannt**

Eines der am häufigsten aufgetretenen Probleme ist, dass das CCM-System die Steuerelemente (Textfelder,Listen, Kontrollfelder...) nicht erkennt, da es sich nicht um Standardkomponenten handelt. Zur Lösung muss in diesem Fall mit Tastensimulation oder Mausklickssimulation gearbeitet werden.

```
LCLICK {„100,100"} 'Maussimulation an Position x:100 y:100
PLAY „{TAB}" `Simulation der Tabulatortaste
```

Die weiteren Möglichkeiten der Maus- bzw. Tastensimulation können im Dokument „Itool Referenz" nachgelesen werden. Es ist jedoch zu beachten, dass je nach Tastaturtreiber nicht alle Tasten unterstützt werden (z.B. linke Windowstaste)

- **Checkbox Problem (Kontrollkästchen)**
  Bei Checkboxen ist es möglich, dass durch die Abwahl einer Checkbox nachfolgende Boxen unsichtbar werden können und somit nicht mehr vom Interpreter en-/disabled werden können. Die Lösung hierfür besteht darin, eine bedingte Ausführung der De-/Aktivierung des Checkbox Kommandos (im Itool-Skript) auszuführen.
  Angenommen es stehen zwei Checkboxen mit der Bezeichnung „Checkbox 1" und „Checkbox 2" zur Auswahl, die beide durch die Adminparameter (CHK1 und CHK2) gesteuert, deaktiviert werden sollen. So sieht der Itool Skriptcode folgendermassen aus:

```
{„Fenstertitel"}
[
     CHECK {„Checkbox 1"} <CHK1>."OFF" OFF
     CHECK {„Checkbox 1"} <CHK1>."ON" ON
     CHECK {„Checkbox 2"} <CHK1>."OFF" OFF
     CHECK {„Checkbox 2"} <CHK1>."ON" ON
]
```

Die Checkbox mit der Bezeichnung „Checkbox 2" ist aber nicht mehr zu de-/aktivieren, sobald die Checkbox mit der Bezeichnung „Checkbox 1"

deaktiviert ist. Dieses Problem kann durch bedingte Skriptausführung gelöst werden:

```
{„Fenstertitel“}
[
     CHECK {„Checkbox 1“} <CHK1>.“OFF“ OFF
     CHECK {„Checkbox 1“} <CHK1>.“ON“ ON
     CHECK {„Checkbox 2“} !<CHK1>.”OFF” & <CHK2>.“OFF“ OFF
     CHECK {„Checkbox 2“} !<CHK1>.”OFF” & <CHK2>.“ON“ ON
]
```

- **Skript bleibt stehen**

  Ein weiteres Problem besteht darin, dass es bei der Ausführung das Skript hängen bleibt und es den Anschein hat, das System könnte den geöffneten Dialog (Fenster) nicht erkennen. Es wird eine Fehlermeldung „Can not execute action in current window“ angezeigt.

  Für dieses Problem gibt es zwei Möglichkeiten.

  1.Möglichkeit:

  Zum einen ist es möglich, dass das System die auszuführende Aktion (besonders oft bei „scheinbaren“ Checkboxen) nicht ausführen kann, da es sich nicht um Standard Windows Komponenten handelt.

  ```
  CHECK {„Checkbox 1“} <CHK1>.“ON“ ON
  ```

  Als Lösung für diese Möglichkeit, kann mittels eines simulierten Mausklicks z.B. die oben erwähnte Checkbox doch aktiviert werden bzw. getestet werden ob darin das Problem bestand.

  ```
  LCLICK {„100,100“} <CHK1>.“ON“
  ```

  2.Möglichkeit

Das Skript bleibt nach der Ausführung des letzten Befehls in einer [Window-Section] stehen. Die Lösung für dieses Problem könnte die vorher erwähnte Möglichkeit sein, oder es müssen Marker eingefügt werden, da es ansonsten möglich ist, dass das System aufgrund selber Fensterbeschriftungen immer den selben Teil des Skripts aufrufen möchte, jedoch die Auszuführende Aktion nicht möglich ist. Der Interpreter kann aufgrund der selben Fenstertitel nicht entscheiden welchen Abschnitt er ausführen soll und nimmt daraufhin per default immer das erste zutreffende.

```
{„Fenstertitel"}
[
     CHECK {„Checkbox 1"} <CHK1>."OFF" OFF
]

{„Fenstertitel"}
[
     BUTTON {„OK"}
]
```

Durch das Einfügen von so genannten Markern, kann diese Problem behoben werden. Bezogen auf obiges Beispiel sähe dies folgendermassen aus wenn zuerst der erste Dialog erscheinen muss und anschliessend der zweite:

```
//erster Dialog
DIALOG1:{„Fenstertitel"}
[
     CHECK {„Checkbox 1"} <CHK1>."OFF" OFF
]

//zweiter Dialog
DIALOG2:{„Fenstertitel"} DIALOG1 & !DIALOG2
[
     BUTTON {„OK"}
]
```

Ist man sich nicht sicher welche der beiden Möglichkeiten für das Problem zuständig sein könnte, so kann man durch einfaches simulieren einen Mausklicks feststellen woran es liegen könnte. Für obiges Bespiel sähe dies folgendermassen aus:

```
Fenstertitel"}
[
     LCLICK {"100,100"}
     CHECK {„Checkbox 1"} <CHK1>."OFF" OFF
]

{„Fenstertitel"}
[
     LCLICK {"100,100"}
     BUTTON {„OK"}
]
```

Mit dieser Erweiterung kann nun festgestellt werden um welches Problem es sich handelt.

- **Skript bricht mit Fehlermeldung ab**

  Bei der Ausführung des Itool (if) Skripts bricht das System dieses mit folgender Fehlermeldung ab:

```
C:\CCM_TEMP\OIT>wininter.exe sysexten.if          wininter
v2.1.5.62 (RELEASE Version) (compiled May 20 2003 at
00:24:14)      (C) 1997-2003 On Technology Corp., Waltham,
MA      Fatal Internal Error: Creating error file.  Unable
to continue.            Error creating new log file
<c:\_integra\pilot\creat01e.010\>. Error Code: 3
```

  Das Problem bestand darin, dass als erstes ein Kommando ausgeführt werden sollte, vor welchem jedoch ein Tabulator eingefügt war.

3.4.2 Spezielle Probleme bei Paketen

- **Positionsproblem**

  Bei der Skriptierung des Pakets „ABB Operate IT Create System with Extensions" trat das Problem auf, dass das CCM-System die zur Auswahl stehenden Checkboxen nicht erkennen konnte. Zwar kann dies wie oben beschrieben (Punkt 3.3.1) durch die Simulation eines Mausklicks behoben werden. Bei diesem Paket jedoch ist die Anzahl der Checkboxen nicht konstant, sondern kann von Fall zu Fall variieren (maximal sechs). Dadurch

entstand ein weiteres Problem. Da im voraus nicht festgestellt werden kann welche Erweiterungen installiert wurden, und davon abhängig welche Checkboxen vorhanden sind, kann nicht gesagt werden an welcher Stelle im Fenster ein Mausklick erfolgen soll.

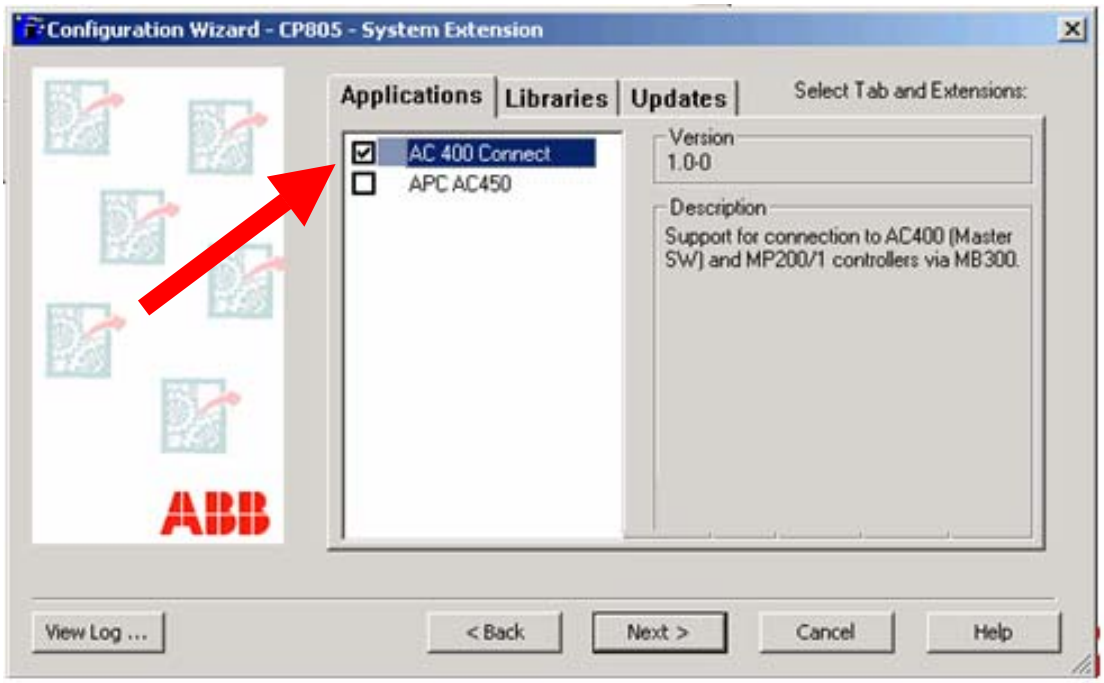


Abbildung 40 - Positionsproblem (Extensions)

Zur Lösung dieses Problems, wurden zuerst für die verschiedenen Erweiterungen, Flags in einer Textdatei gespeichert. So wurde wenn zum Beispiel die Erweiterung AC400 installiert war in einer Textdatei namens ac400.txt eine 1 eingetragen, im ungekehrten Falle eine 0. somit war schon das erste Problem gelöst. Mit dieser Art konnte nun festgestellt werden, ob die jeweiligen Erweiterungen installiert waren.

Die Lösung der Positionsbestimmung war nicht so trivial. Hierfür wurde ein Visual Basic Skript programmiert, welches die vorher beschriebenen Textdateien einliest, und anhand der Flags auswertet an welcher Stelle im Fenster für die jeweilige Extension ein Mausklick erfolgen soll. Natürlich musste hierbei auch ausgewertet werden, welche Erweiterungen überhaupt in das System eingebunden werden sollten. Damit das CCM-System auch auf

das Ergebnis der Positionsbestimmung reagieren konnte, wurde sechs Umgebungsvariablen definiert, die, wenn die jeweilige Erweiterung installiert ist und eingebunden werden sollte gesetzt werden. So konnte im Anschluss das Itool Skript in Abhängigkeit von diesen Variablen die Mausklicks an der jeweiligen Position simulieren.

Der komplette Quellcode ist im Anhang B (Quellcode position.vbs) zu sehen.

- **Anwendung als bestimmter Benutzer ausführen**
  Bei der Installation verschiedener Pakete, trat das Problem auf, dass die Installationsroutine unter einem bestimmten Benutzer ausgeführt werden musste. Windows stellt dafür das runas.exe Programm zur Verfügung. Details zu diesem Programm, beziehungsweise zu den Parametern, können in der Microsoft Windows Hilfe nachgelesen werden. Leider kann diesem Tool nur der Benutzername und nicht dass zu verwendende Benutzerpasswort per Aufruf mitgegeben werden. Somit musste eine Lösung für dieses Problem gefunden werden.

  Beispiel Aufruf:

```
runas.exe /user:Administrator x:\install\setup.exe
```

  Die Lösung dieses Problems, war wiederum ein Visual Basic Skript (runas.vbs), welches die übergebenen Argumente in einer for-Schleife einliest. Daraufhin wird ein neues Shell Objekt erzeugt, und diesem Objekt mitgeteilt es solle das runas.exe Programm mit den Parametern Benutzername und dem Namen des ausführenden Programms starten. Anschliessend werden das Passwort und zum abschliessen der Eingabe die ENTER Taste mittels des Visual Basic Kommandos sendkeys an die Kommandozeile gesendet. Somit war es nun möglich, eine Anwendung oder ein Setupprogramm mit der Angabe des Benutzer und dem zugehörigen Passwort zu starten, ohne dabei von Hand eingreifen zu müssen.

Syntax : `runas.vbs <username> <password> <command>`

Beispiel Aufruf:

```
cscript.exe runas.vbs Administrator AdminPasswort
x:\install\setup.exe
```

Anmerkung: *cscript.exe* startet dabei den Windows Skripthost, der zur Ausführung von Visual Basic Skripts in der Kommandozeile benötigt wird (Details siehe Windows Hilfe).

Der komplette Quellcode ist im Anhang C (Quellcode runas.vbs) zu sehen.

- **Treiberproblem**

  Bei der Installation des Pakets ABB Operate IT PPA 2.1-2 incl. SP1 (ppa01e.021) sollte ein spezieller Treiber für ein RTA-Board installiert werden. Diese PCI-Karte wird für die Ankopplung des Feldbusses der Kraftwerkssteuerung benötigt. Bei der Installation des Treibers hierfür, kam es jedoch zu dem Problem, dass das Setup diesen nicht installieren konnte. Als erste Vermutung für dieses Problem, kamen fehlende Berechtigungen während der Setupphase in Betracht. Nach längerer Analyse, stellte sich jedoch heraus, dass es sich nicht um fehlende Berechtigungen handeln konnte. Der Benutzer (smeclnt) unter dem das vollautomatische Setup ausgeführt wird, hat in der Zeit in der das Setup ausgeführt wird volle Administratorrechte. Somit konnte es nicht an den fehlenden Berechtigungen liegen.

  Durch ausprobieren wurde festgestellt, dass das Setup und gleichzeitig die darin enthaltene Treiberinstallation durch starten des Setups mittels des Windows Start -> Ausführen Dialogs ohne Probleme durchlief.

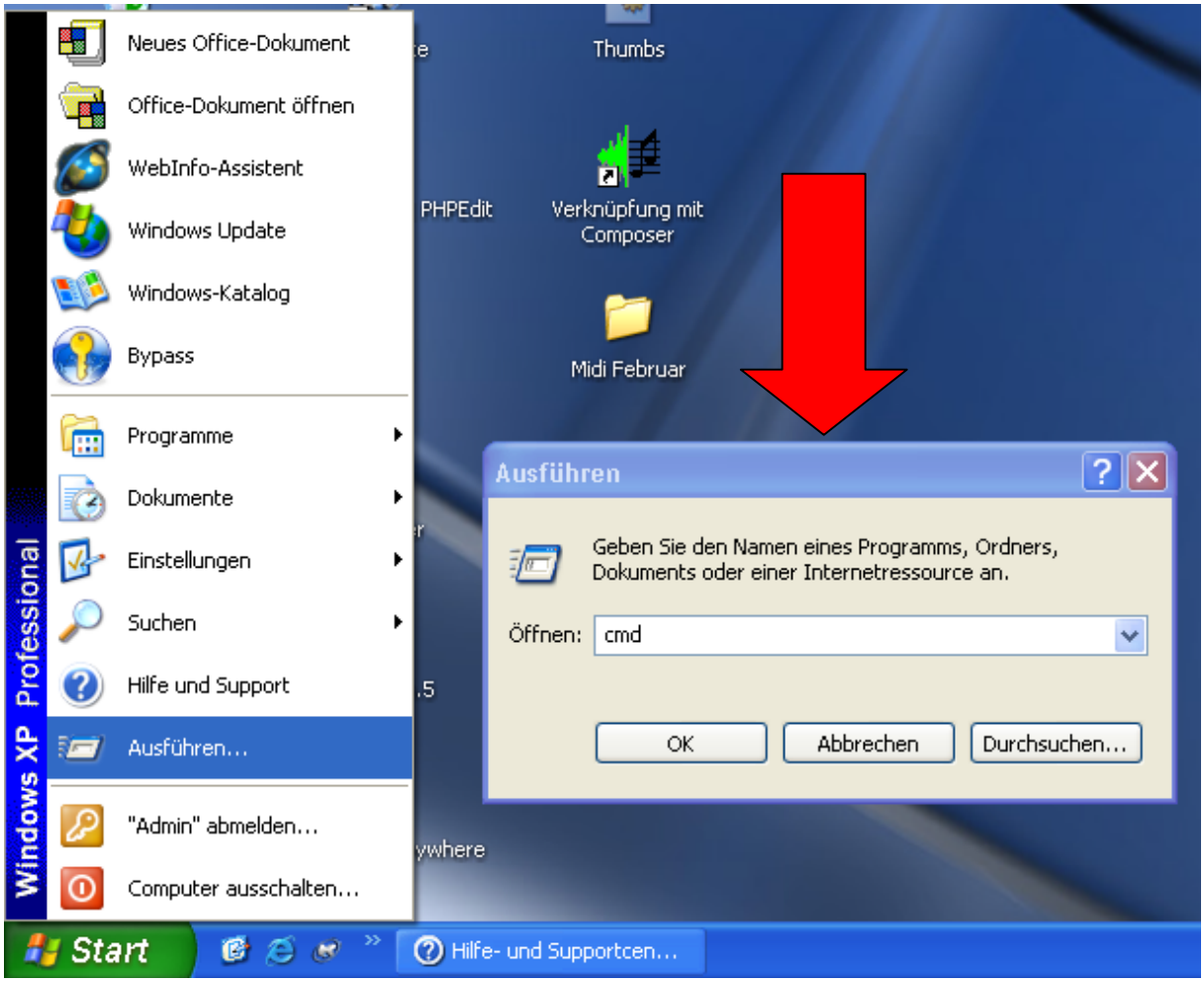


Abbildung 41 - Windows Start -> Ausführen

Somit war das Problem identifiziert. Nun musste eine Lösung gefunden werden, diesen Dialog aufzurufen. Der erste Versuch dies mittels der Simulation der Tastenkombination linke Windows + r Taste den Dialog aufzurufen funktionierte leider nicht. Auch die Simulation mittels Mausklicks brachte keinen Erfolg. Dies ist darauf zurückzuführen, dass die Windows Oberfläche nicht wie sonstige Dialoge bzw. Fenster vom CCM-System gehandhabt werden kann.

Daraufhin wurde die Windows Programmierschnittstelle (API – Application Programming Interface) nach einer Visual Basic Funktion für den Aufruf dieses Dialogs durchsucht. Visual Basic daher, da es hierbei, im Gegensatz zu Visual C++, um ein vielfaches einfacher ist, die Windows Funktionen zu verwenden.

Die erste gefundene Möglichkeit war mittels der FileRun – Methode, aus der Bibliothek Microsoft Shell Controls And Automation, den Dialog zu öffnen. Dazu wurde folgender Code in Visual Studio eingegeben und daraus dann eine ausführbare (exe-Datei) erzeugt.

```
Private Sub Form_Load ()
     Dim MyShell As New Shell
     MyShell.FileRun
     Unload Form1
End Sub
```

Leider rief diese Funktion zwar den Dialog „Ausführen“ auf, jedoch konnte der Treiber auch weiterhin nicht installiert werden.

Auf der Suche nach einer Erklärung was der Unterschied zwischen dem Code und dem manuellen Aufruf des Dialogs sein konnte, wurde eine von Microsoft undokumentierte Funktion (SHRunDialog) zum Aufruf dieses Dialogs gefunden.
Daraufhin wurde diese Funktion anstatt der oberen in Visual Basic eingebaut und wieder eine ausführbare Datei erzeugt.

```
Option Explicit

Private Declare Function SHRunDialog Lib "shell32" Alias _
     "#61" (ByVal hWnd As Long, ByVal hIcon As Long, _
     ByVal sPath As String, ByVal sTitle As String, _
     ByVal sPrompt As String, ByVal uFlags As Long)As Long

Dim hWnd as Long

Private Sub Form_Load ()
     SHRunDialog hWnd,0,0,vbNullString,vbNullstring,0
     Unload Form1
End Sub
```

Auch diese führte zum Aufruf des Ausführen Dialogs, jedoch konnte der Treiber auch mit dieser Art nicht installiert werden.

Daraufhin wurde nach weiteren Möglichkeiten gesucht, diesen Dialog aufzurufen. Als nächste Möglichkeit wurde folgender Code verwendet, bei dem dies über Objekte gehandhabt wird.

```
Private Sub Form_Load ()
     Dim oShell As Object
     Set oShell = GetObject ("", "Shell.Application")
     If ObjPtr(oShell) Then
          oShell.FileRun
     else
          oShell = CreateObject("Shell.Application")
          oShell.FileRun
     End if
     Unload Form1
End Sub
```

Mit dieser Methode funktionierte schliesslich die Installation des Treibers für das RTA-Board. Somit war das Problem der Treiberinstallation gelöst. Der Grund warum es mit den anderen Funktionen nicht funktionierte, konnte wegen fehlender bzw. nicht tief genug gehender Dokumentation leider nicht herausgefunden werden.

Der komplette Quellcode ist im Anhang D (Quellcode startrun.exe) nochmals aufgeführt.

- **RunOnce Problematik**

  Bei der Installation diverser Anwendungen ist es möglich, dass diese einen Eintrag in der Registry, unter RunOnce und RunOnceEx machen. Dies hat zur Folge, dass beim nächsten Anmelden nach einem Neustart, dass unter diesem Schlüssel angegebene Kommando ausgeführt wird.
  Hier besteht nun das Problem, dass es möglich ist dass die Dialoge bereits vor dem erscheinen der Windows Oberfläche erscheinen und eine Eingabe erwarten. Somit besteht keine Möglichkeit die erscheinenden Fenster im Development Studio aufzuzeichnen.

  Für dieses Problem kann die im Anhang E (runonce.bat) aufgeführte Batchdatei verwendet. Diese Batchdatei muss, nachdem der erste Teil der

Installation fertig gestellt ist und noch vor dem Neustart, mit dem Parameter *get* aufgerufen werden. Dieser Aufruf der Batchdatei sorgt dafür, dass die in der Registry unter dem RunOnce und RunOnceEx eingetragenen Befehle bzw. Programmaufrufe gespeichert werden.

```
call runonce.bat get
reboot 1
:REBOOT1
```

Nach dem reboot werden diese Kommandos, durch den erneuten Aufruf der runonce.bat, diesesmal jedoch mit dem Parameter *run*, ausgeführt.

```
call runonce.bat run
```

Somit hat man anschließend die Möglichkeit die nach dem Neustart erscheinenden Fenster im Development Studio aufzuzeichnen. Die oben beschriebenen Aufrufe der Batchdatei müssen bei dem beschriebenen Problem jedoch weiterhin in der config.swp eingetragen bleiben.

4. Sonstige Tätigkeiten

4.1 Integration bestehender Clients

Im Folgenden werden die Vorraussetzungen, Möglichkeiten und Probleme bei der Integration bestehender Clients beschrieben. *Quellen [13][14][15][16]*

4.1.1 Autodiscover (OnWards)

Da in der aktuellen Umgebung der Firma bereits zahlreiche installierte Rechner vorhanden sind, war ein wichtiger Aspekt, dass bestehende Rechner mit dem System integriert werden können. Hierzu liefert der Hersteller sein Tool OnWards mit. Mit diesem besteht die Möglichkeit, Clients in die Datenbank einzubinden und deren Zustand zu erfassen.

Dazu müssen zuerst die zu erkennenden Applikationen dem System bekannt gemacht werden. Hierzu sind Einträge in einer Konfigurationsdatei auf dem Server in folgender Art gemacht werden:

```
Acrobat Reader 5.0 = aread01e.050
<Name des Anwendung> = <PaketId>
```

Der Name der Anwendung, muss dabei wie in der Windows Registry eingetragen sein. Ansonsten kann das System die installierte Anwendung nicht erkennen. Dabei wird jedoch nicht die komplette Registry durchsucht, sondern nur der Teil in dem die Deinstallationseinträge vorhanden sind. Unter dem jeweiligen Schlüssel muss ein Eintrag DisplayName vorhanden sein, der den Namen der Anwendung enthält. Diese sind so z.B. unter Windows 2000 unter

```
HKEY_LOCAL_MACHINE
|
|--> SOFTWARE
     |
     |--> Microsoft
```

```
|
|--> Windows
     |
     |-->CurrentVersion
          |
          |-->Uninstall
               |
               |--> DirectX
                    |
                    |-->DisplayName="DirectX 8.1"
```

zu finden.

4.1.2 Probleme Autodiscover

Registryeinträge

Die oben beschriebenen benötigten Einträge in der Registry sind sicherlich ein grosser Nachteil des Systems, da nur ein Teil der Applikationen auch wirklich einen Eintrag in der Windows Registry macht, beziehungsweise gar kein Deinstallationsroutine besitzt (z.B. DirectX9) und somit auch nicht erkannt werden kann.

Daher müssen die bereits integrierten Anwendungen darauf überprüft werden, ob diese einen Eintrag im Uninstall Zweig der Registry tätigen. Ist dies nicht der Fall so muss ein entsprechender Eintrag erzeugt werden.

Dies kann durch folgende Befehlszeile in der config.swp Datei erfolgen. Dabei wird der Eintrag aus der angegebenen Datei (autodisc.reg) in die Registry eingetragen.

```
regdat3 -i c:\_integra\sw\paketname\autodisc.reg
```

Inhalt autodisc.reg:

```
REGEDIT4

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\DirectX]
"DisplayName"="DirectX 8.1"
```

Wird das Paket wieder deinstalliert, so muss natürlich auch der entsprechende Eintrag in der Registry wieder entfernt werden. Hierzu kann die gleiche Datei (autodisc.reg) verwendet werden. Lediglich der Aufruf hat einen anderen Parameter.

```
regdat3 -d c:\_integra\sw\paketname\autodisc.reg
```

Somit ist das Problem dass der Autodiscover Service die Anwendungen nicht erkennt gelöst.

Applikationen wurden nicht erkannt

Zu Beginn der Integration wurden keine Applikationen erkannt, obwohl diese wie erforderlich in der Datei *applications.cfg* eingetragen waren. Auch diverse Telefonate mit der Hotline konnten das Problem nicht lösen. Nach mehreren Versuchen stellte sich heraus, dass das Problem darin bestand, dass die Einträge für Applikationen welche zum Zeitpunkt des Startens des Autodiscover Service nicht in der *applications.cfg* eingetragen sind, gemacht werden können (damit diese auch erkannt werden) daran Schuld aren. Nach dem entfernen dieser Einträge wurden die in der *applications.cfg* eingetragenen Anwendungen erkannt.

4.2 Zusammenfassung von Konfiguration

Das System bietet die Möglichkeit so genannte Profile anzulegen. Dabei handelt es sich um eine Zusammenfassung einer Konfiguration. Hierbei besteht die Möglichkeit spezielle Hardware- und Softwareprofile anzulegen. Dies hat den Vorteil dass einem Client nicht jedes Paket einzeln zugewiesen werden muss, sondern ein komplettes Profile, dass das Betriebssystem inklusive aller Anwendungen enthält, mit einem Mausklick zugewiesen werden kann.

In der folgenden Abbildung ist solch ein erstelltes Softwareprofil zu sehen. Dieses beinhaltet die Installation des Betriebssystems, sowie aller benötigten Anwendungen (MS Office, Visual Basic, Acrobat Reader usw.) mit bereits vordefinierten Parametern.

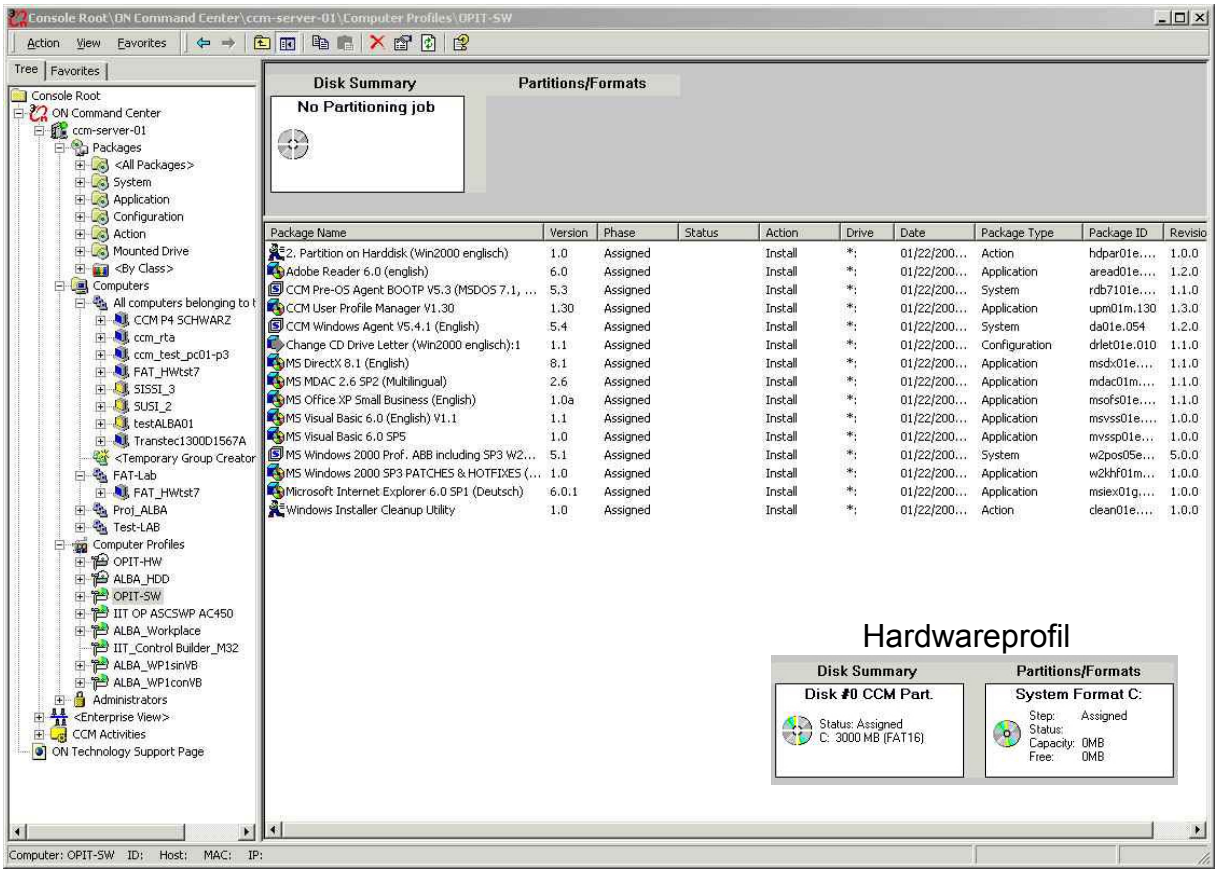


Abbildung 42 - CCM Hardware- und Softwareprofil

Zu einem solchen Softwareprofil kann noch ein Hardwareprofil definiert werden. In diesem Profil ist jedoch lediglich die Partitionierung der Festplatte definiert. Die Darstellung eines solchen Profils ist in obiger Abbildung 21 im rechten unteren Teil des Fensters zu sehen.

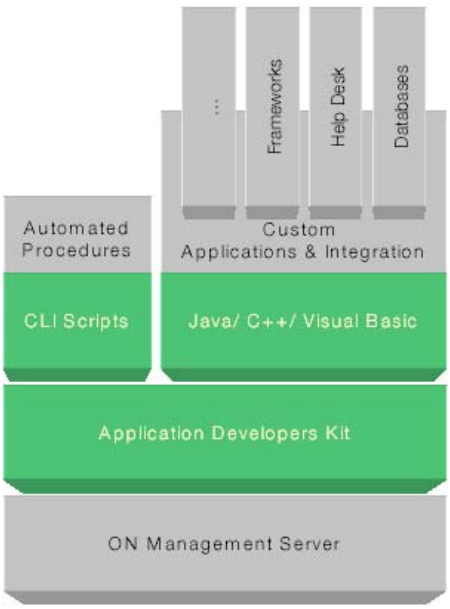
4.3 Zugriffsberechtigungen

Mit dem System besteht die Möglichkeit bestimmte Benutzer bzw. Benutzergruppen anzulegen, die nur bestimmte Rechte haben. Dies ist vergleichbar mit der Windows NT/2000/XP Benutzerverwaltung. Da zum jetzigen Zeitpunkt noch nicht feststand wie

die zukünftige Betreuung des Systems aussieht, wurden ausser den Administratoren Accounts, noch ein Account angelegt, der ausschliesslich Pakete installieren darf.

4.3 Hinweismeldung bei Fehler

Da bei der Vorführung des Systems aufgrund eines Netzwerkausfalls die Installation eines Rechners nicht klappte, sollte eine Möglichkeit zum Absenden einer Textmitteilung via SMS (Short Message Service) gesucht werden. Dies wird vom CCM-System leider nicht unterstützt.
Zur Lösung dieses Problems wurden zwei Möglichkeiten gefunden. Die erste besteht darin, mittels des von OnTechnology angebotenen Tools OnCommand Alerter eine E-Mail an eine bestimmte Adresse zu schicken. Da dies jedoch noch nicht die Möglichkeit des SMS-Versands ermöglicht, müsste diese E-Mail an den CCM-Server oder ein Gateway gesendet werden, welches dann diese E-Mail in eine SMS umwandelt. Diese Funktionalität ist Standardmässig bereits im Windows 2000 Server Betriebssystem integriert. Eine weitere Möglichkeit der Umwandlung besteht darin, diese E-Mail an einen Webservice Anbieter (z.B. eMail2SMS) zu senden, welcher diese dann per SMS weiterschickt. Ein solcher Service ist für eine monatliche Grundgebühr von CHF 39,- (inklusive 50 SMS, jede weitere 12 Rp./SMS) zum Beispiel unter www.infomotion.ch zu beziehen.



Die zweite Möglichkeit besteht darin, mittels dem OnCommand ADK (Applikation Developers Kit) auf das CCM-System bzw. die CCM-Datenbank zuzugreifen. Dabei handelt es sich um eine Entwicklungsumgebung, die vollen Zugriff auf die Programmierschnittstellen (API), die offene Schnittstelle Command Line Interface (CLI) und Itool bietet. Dies ist sicher die flexiblere Lösung für dieses Problem und bietet auch für die Zukunft durchaus mehr Möglichkeiten.

Abbildung 43 - CCM Schnittstellen

Dadurch kann mittels Java, C++ oder Visual Basic bei einem Fehler eine SMS versand werden. Hierzu kann dann z.B. durch die Verwendung einer Java Bibliothek zum versenden von SMS die gewünschte Funktionalität erreicht werden. Folgende Bibliotheken könnten dabei zum Einsatz kommen:

- ***jSMSEngine*** – kostenloses Java API Paket zum versenden und empfangen von SMS Nachrichten mittels Mobiltelefon oder GSM Modem

Abbildung 44 - OnCommand Schnittstellen

- ***SMProLib*** – Java Bibliothek zum versenden von SMS Nachrichten mittels Mobiltelefon das über USB/COM/IrDA mit dem Computer verbunden ist (Preis 100 $)

Die Listenpreis für den Alerter beträgt 4000,- SFR und für das ADK 17500,- SFR.

5.Fazit

Die Firma OnTechnology bietet mit ihrem CCM System und den verschiedenen Tools eine gute Komplettlösung für komplexe Softwareverteilung in mittleren und grösseren Unternehmen. Für kleinere Unternehmen ist das System aufgrund des grossen und mit hohen Kosten verbundenen Initialaufwands eher ungeeignet. Aufgrund der Systemarchitektur ist es möglich dass mehrer Clients gleichzeitig aufgesetzt werden können, ohne dass allzu grosse Netzwerkbelastungen auftreten werden. Von einem zentralen Server aus können die Daten auf abgesetzten Depot Servern (z.B. in verteilten Netzen) repliziert werden. Somit ist auch das Problem der Softwareverteilung in verteilten Netzen gelöst. Durch diese Möglichkeit kann auch die Belastung der Verbindung zwischen dem zentralen Server und den Depot Servern klein gehalten werden, da die Depotdaten mittels CD versandt werden können. Somit müssen nur die Skript- und Konfigurationsdateien über eine Verbindung zwischen den Servern (Modem, Internet …) übertragen werden. Mit Hilfe der Log-Files, die im Command Center betrachtet werden können, besteht jederzeit die Möglichkeit den Status der Verteilung zu überwachen, steuern und auszuwerten. Auch das Absenden einer Meldung beim Auftreten eines Fehlers ist sicherlich eine sehr nützliche Funktion. Die Erstellung der Skripte mit dem Development Studio ist bei den meisten Anwendungen sehr unproblematisch, sofern man erst einmal einen Einblick in das System erhalten hat und anschliessend weiss worauf geachtet bzw. wo nach Fehlern gesucht muss. Eine sehr gute Funktion ist sicherlich, dass Installationspakete zu Konfigurationen zusammengefasst werden können, womit einiges an Arbeit, im Gegensatz zum separatem Zusammenstellen, erspart werden kann. Die Möglichkeit bestehende Clients, welche bereits ein Betriebssystem und diverse Anwendungen installiert haben, in das System zu integrieren erspart sicherlich viel Arbeit und bietet die Möglichkeit eine Inventarisierung der vorhandenen Hardware und Software durchzuführen. Wichtig ist sicherlich auch dass nicht jeder Zugriff auf das System haben soll, oder nur bestimmte Aufgaben erledigen darf. Mit dem integrierten Rollenkonzept stellt dies auch kein Problem mehr dar. Die im Rahmen eines Wartungsvertrags zur Verfügung gestellten Skripte zur Softwareverteilung helfen, da diese nicht mehr selbst generiert werden müssen, den Aufwand zu reduzieren.

Auf der anderen Seite muss der extrem hohe Initialaufwand für die Einführung des Systems gesehen werden. Dabei entstehen nicht nur hohe Kosten für die Erstellung der Skripte, sondern auch für das System selber. Gern wird dabei auch der mit der Einführung eines solchen Systems verbundene Zeitaufwand unterschätzt. Der Preis für ein solches System ist von der Anzahl der Clients und Server abhängig. Aufgrund dieser Nachteile ist das System für kleine Netze, aufgrund des grossen Gesamtaufwands eher ungeeignet. Da jeder Client, während er sich die Installationsdateien besorgt, eine eigene Verbindung zum Server herstellt (kein Multicast), kann eine unnötige Netzwerkbelastung entstehen. Der aus meiner Sicht grösste Nachteil für den Benutzer ergibt sich jedoch aus der Tatsache, dass der Client auf dem ein Update- oder Installationsvorgang durchgeführt werden soll, für die komplette Zeit gesperrt ist. Der Benutzer hat somit keine Möglichkeit während dieser Phase weiterzuarbeiten.

Im Grossen und Ganzen kann gesagt werden, dass man mit diesem System ein komplexes und sehr gutes Werkzeug zur automatischen Softwareverteilung erhält.

Anhang A – Eigene Pakete

Paketname	Paket ID	Beschreibung	Besonderheiten
Standardanwendungen			
2. Partition on Harddisk (Win2000 englisch)	hdpar01e.011	erstellt eine 2.Partition	Problem dass simulierte Eingaben teilweise nicht funktionierten
Acrobat Reader 4.0 (English)	aread01e.040	installiert Acrobat Reader 4	---
Acrobat Reader 5.0 (Deutsch)	aread01g.050	installiert Acrobat Reader 5	---
Acrobat Reader 5.0 (English)	aread01e.050	installiert Acrobat Reader 5	---
Change CD Drive Letter (Win2000 englisch)	drlet01e.010	ändert den CD-Laufwerksbezeichnung	Problem dass simulierte Eingaben teilweise nicht funktionierten
McAfee Antivirus 6.0 (English)	mcafe01e.060	installiert McAfee 6	---
McAfee Antivirus 7 (English)	mcafe01e.070	installiert McAfee 7	2 Versionen interne mit Lizenz und externe ohne
McAfee AntiVirus Update ver.6 & 7 (English)	u_vsc01e.010	spielt Virusupdate ein	---
MS Windows 2000 Professional ABB incl. SP3 (English)	w2pos05e.050	installiert Windows 2000 eng.	Treiber,SP3 bereits integriert
MS Windows 2000 Professional ABB inkl. SP3 (Deutsch)	w2pos05g.050	installiert Windows 2000 dt.	Treiber,SP3 bereits integriert
Pc Anywhere 10.5 (English)	pcany01e.105	installiert PcAnywhere 10.5	5 Skripte für die Optionen Professional,Individual,Remote,Host,Lan)
WinZip 8.1 (english)	wzip01e.081	installiert WinZip 8.1	---
Ahead Nero 6.0 (Deutsch)	nero01g.060	installiert Nero 6.0	---
MS Office 2000 SB (English)	msofs01e.090	installiert Office 2000 Small Business	3 Installationskonfigurationen (Complete,Custom,Default) die dann kopiert werden & ausgeführt (jeweils spez. MSI File)
MS Office XP SB (English)	msofs01e.010	installiert Office XP Small Business	3 Installationskonfigurationen (Complete,Custom,Default) die dann kopiert werden & ausgeführt (jeweils spez. MSI File)
Windows Installer Cleanup Utility (English)	clean01e.010	installiert Windows Cleanup Utility	Benötigt, damit OFFICE XP nach Deinstallation wieder erneut installiert werden kann
MSI Wizard 2.0 NT/2000/XP (English)	msi01e.020	installiert MSI Wizard 2.0	---
LeechFTP 1.3 (Multilingual)	leftp01g.013	installiert LeechFTP	Multilingual -> exe die OS Daten ausliest und dann in batch schreibt

MS Windows 2000 SP3 HOTFIX & PATCHES(Englisch)	w2khf01e.010	installiert Windows 2000 Hotfixes und Patches ab Service Pack 3	• Batchfile für die Installtion der Patches werden generiert • Patchsel.bat – enthält die Variablen der ausgewählten Patches • Abfrage.bat – wird an patchsel.bat angehängt und daraus dann die select.bat erzeugt • Qchain.exe
MS Windows 2000 Install PATCH/HOTFIX	w2kpa01e.010	installiert den per Dateinamen angegebenen Patch/Hotfix	---
McAfee Antivirus 7 INTERNAL USE ONLY (Englisch)	mcafi01e.070	installiert McAfee 7	---
Winzip 7.0 (deutsch)	wzip01g.070	installiert Winzip 7.0	---
MS DirectX 8.1(Englisch)	msdx01e.081	installiert DirectX 8.1	kann wieder deinstalliert werden (vorher-nachher analyse)
MS DirectX 9.0b (Mutilingual)	msdx01m.090	installiert DirectX 9.0	kann wieder deinstalliert werden (vorher-nachher analyse)
CCM User Profile Manager	upm01m.130	installiert User Profile Manager	• Da gateway bei Installation nicht gesetzt wird wird dies vom upm erledigt • Patch MS03-026 und Ms03-039 werden installiert
Pc Anywhere 9.2 (English)	pcany01e.092	installiert PcAnywhere 9.2	---
Configure/Modify DNS	c_dns01e.010	konfiguriert/modifiziert die DNS einstellungen der Netzwerkkarten	Netsh benutzt, kein Itool Skript
Windows NT SP6a HOTFIX&PATCHES(Multilingual)	wnthf01e.010	installiert Windows NT Hotfixes und Patches ab Service Pack 6a	---
MDAC 2.6 SP2 (Multilingual)	mdac01m.026	installiert Microsoft MDAC Service Pack	Musste herausfinden welche OS-Sprache installiert ist (getosver.exe)
MS Visual Basic 6 (English)	msvss01e.060	installiert Visual Basic 6	2 Installationsskripte, da bei Installation ein reboot erforderlich ist
MS Visual Basic 6 SP5 (English)	mvssp01e.050	installiert Service Pack 5 für Visual Basic 6	---
Netmeeting Windows NT 4.0 (English)	netme01e.031	installiert Microsoft Netmeeting	Profil musste kopiert werden, da verknüpfungen nur unter benutzer smeclnt sichtbar waren

Exceed 7.1 (Multilingual)	exced01m.071	installiert Exceed 7.1	---
Exceed 8.0 (Multilingual)	exced01m.080	installiert Exceed 8.0	Verschiedene Transformationsfiles
Java Runtime Environment 1.3.1 (Multilingual)	sujre01m.013	installiert SUN JRE 1.3.1	---
Office SP/HF/SR	ofsp01e.010	installiert Service Pack und Hotfixes für Office 2000 & XP	Zu installierende Patches unterstützen die auf MS-Seite angegebenen Optionen nicht -> ini-files angepasst
ABB Applikationen			
ABB Operate IT PPA	ppa01e.010	installiert PPA	
ABB Operate IT PPA 2.1-2	ppaup01e.021	installiert PPA Update 2.1-2	
ABB Operate IT PPA 2.1-2 SP1	ppasp01e.010	installiert PPA Service Pack 1	
ABB Operate IT AC400	ac40001e.010	installiert AC400	
ABB Operate IT AC400 2.1-2	ac4up01e.021	installiert AC400 Update 2.1-2	
ABB Operate IT AC400 2.1-2 SP1	ac4sp01e.021	installiert AC400 Service Pack 1	
ABB Operate IT APC AC450	apcac01e.010	installiert APC AC450	Für Positionsbestimmung -> Flag in Datei gesetzt
ABB Operate IT P13 Connect CI	p13co01e.010	installiert P13 Connect	Für Positionsbestimmung -> Flag in Datei gesetzt
ABB Operate IT AC800 Connect	ac80001e.010	installiert AC800 Connect	Für Positionsbestimmung -> Flag in Datei gesetzt
ABB Operate IT Control Builder M Pro 3.2	buildp01e.032	installiert Control Builder M	---
ABB Operate IT OPC Server AC800 3.2	opcsv01e.032	installiert OPC Server für AC800	Prozesse mussten beendet werden damit Setup funktioniert
ABB Operate IT AC100 Connect	ac10001e.022	installiert AC100 Connect	Für Positionsbestimmung -> Flag in Datei gesetzt
ABB Operate IT Create System with Extensions	creat01e.010	erzeugt ein OperateIT System	---
ABB Operate IT PPA Configuration	ppaad01e.010	konfiguriert PPA	Erst hier wird der Admin Account angelegt
ABB Operate IT Soft Controller 3.2	softc01e.032	installiert Soft Controller 3.2	Zur Deinstallation muss der OPC-Server Prozess beendet sein -> killopc.bat
*wurden durch folgende neueren Version ersetzt			
ABB Operate IT PPA 2.1-2 incl. SP1	ppa01e.021	installiert PPA inklusive Update & Service Pack	• Für Positionsbestimmung -> Flag in Datei gesetzt (RNRP) • Bei der ersten

			Installation wird smeclnt als Admin von PPA angelegt, da wundersammerweise der Adminitratoraccount nicht akzeptiert wird (-> Paket PPA Configuration)
ABB AC400 2.1-2 incl. SP1	ac40001e.021	installiert AC400 inklusive Update & Service Pack	• Startrun.exe -> Treiberinstallationsproblem • Für Positionsbestimmung -> Flag in Datei gesetzt
ABB Advabuild CBA 1.0 Program CD1	adcba01e.010	installiert CBA Program	Aufgrund der Installationsoptionen (Application Builder, Bus Configuration Builder…) und der dadurch erforderlichen reboots, mussten für diese für jede Deinstallation ein extra Skript erzeugt werden
ABB Advabuild CBA 1.0 Library CD4	adlib01e.010	installiert CBA Bibliotheken	Aufgrund der Installationsoptionen (AC110-,AC160-Library...) und der dadurch erforderlichen reboots, mussten für diese für jede Deinstallation ein extra Skript erzeugt werden
ABB Advabuild CBA 1.0 FCB Addons	adadd01e.010	installiert CBA Addons	• Batch-File (run.bat) da ansonsten CCM das vorhandene Batchfile nicht starten konnte • Awk konnte nicht auf Datei im Unetrverzeichnis zugreifen -> copy nach root -> modifizieren -> copy ins unterverzeichnis
ABB Advant OPC Server 3.0	adopc03e.010	installiert OPC Server für Advabuild	Ronce.bat Problematik
ABB Advant AS160OS	advos01e.010	installiert AS160 Operator Station	---
ABB Advant OPC Server 3.0 Service Pack 1	adosp01e.010	installiert OPC Server Service Pack	• Ronce.bat Problematik • Prozesse mussten beendet werden, damit SP installiert werden kann • Startrun.exe um Service zu registrieren

			• Registry Modifikation
Progress 3	prog301e.010	installiert Progress 3	5 Skripte aufgrund Installationsanleitung
PCMCIA Drive for Win2k/XP	pcmic01e.010	installiert PCMCIA Laufwerk unter Windows 2000/XP	• Auf Desktop wird Seriennummer abgelegt, da sich das Programm nach der Installation die eingetragene Nummer nicht gemerkt hat • Registrymodifikation
Intellution FIX	infix01e.010	installiert Intellution FIX	Netzlaufwerk gemountet (Net use) da ansonsten Setup nur Fehler brachte
FIX OPC Client			-> in Intellution FIX Paket enthalten

Anhang B – Quellcode position.vbs

```
'position Visual Basic Skript
'Author: Martin Hald ABB Schweiz AG
'Datum: 19.01.2004
'Syntax: position.vbs
'bestimmt anhand der installierten Erweiterungen (PPA) an
'welcher Position der Mausklick im Dialog erfolgen muss

'Deklaration der benötigten Variablen
Dim fso, StreamOut, StreamIn
Dim fp13, fac450, fac800, fac100, fapc, frnrp, pos, search
Dim path
Dim finish

finish = False
pos = 0

'Dateinamen der Dateien die die Flags enthalten
fp13 = "P13.txt"
fac100 = "ac100.txt"
fac400 = "ac400.txt"
fapc = "apc.txt"
fac800 = "ac800.txt"
frnrp = "rnrp.txt"
search = "search.txt"

'anlegen eines Streaming Objekts (FileSystemObject) zum
'einlesen und schreiben von Dateien
Set fso = CreateObject("Scripting.FileSystemObject")
Set StreamOut = fso.CreateTextFile("erg.bat")
Set StreamIn = fso.OpenTextFile(search)

'einlesen des Dateiinhalts und entfernen von ueberfluessigen
'Zeichen
s = StreamIn.ReadLine()
s = UCase(s)
s = Trim(s)


'Wscript.Echo "RNRP Suche:" & s & " X:"& X1 & " Position:" &
'pos

Set StreamIn = fso.OpenTextFile(fp13)
X4 = StreamIn.ReadLine()
X4 = Trim(X4)

'Suchvorgang für die Positionsbestimmung
'dabei wird geprueft ob die Erweiterung installiert ist und
'daraus dann die Position für den Mausklick bestimmt
```

```
If Not finish = True Then
    If Not StrComp("P13", s) = 0 Then
        If StrComp(X4, "1") = 0 Then
            pos = pos + 1
        End If
    Else
        If StrComp(X4, "1") = 0 Then
            pos = pos + 1
        End If
        finish = True
    End If
End If

Set StreamIn = fso.OpenTextFile(fac800)
X5 = StreamIn.ReadLine()
X5 = Trim(X5)

If Not finish = True Then
    If Not StrComp("AC800", s) = 0 Then
        If StrComp(X5, "1") = 0 Then
            pos = pos + 1
        End If
    Else
        If StrComp(X5, "1") = 0 Then
            pos = pos + 1
        End If
        finish = True
    End If
End If

Set StreamIn = fso.OpenTextFile(fac400)
X2 = StreamIn.ReadLine()
X2 = Trim(X2)

If Not finish = True Then
    If Not StrComp("AC400", s) = 0 Then
        If StrComp(X2, "1") = 0 Then
            pos = pos + 1
        End If
    Else
        If StrComp(X2, "1") = 0 Then
            pos = pos + 1
        End If
        finish = True
    End If
End If

'Wscript.Echo "AC400 Suche:" & s & " X:"& X1 & " Position:" &
pos
```

```
Set StreamIn = fso.OpenTextFile(fapc)
X3 = StreamIn.ReadLine()
X3 = Trim(X3)

If Not finish = True Then
    If Not StrComp("APC", s) = 0 Then
        If StrComp(X3, "1") = 0 Then
            pos = pos + 1
        End If
    Else
        If StrComp(X3, "1") = 0 Then
            pos = pos + 1
        End If
        finish = True
    End If
End If

Set StreamIn = fso.OpenTextFile(frnrp)
X1 = StreamIn.ReadLine()
X1 = Trim(X1)

If Not finish = True Then
    If Not StrComp("RNRP", s) = 0 Then
        If StrComp(X1, "1") = 0 Then
            pos = pos + 1
        End If
    Else
        If StrComp(X1, "1") = 0 Then
            pos = pos + 1
        End If
        finish = True
    End If
End If

Set StreamIn = fso.OpenTextFile(fac100)
X6 = StreamIn.ReadLine()
X6 = Trim(X6)

If Not finish = True Then
    If Not StrComp("AC100", s) = 0 Then
        If StrComp(X6, "1") = 0 Then
            pos = pos + 1
        End If
    Else
        If StrComp(X6, "1") = 0 Then
            pos = pos + 1
        End If
        finish = True
    End If
```

```
End If

'Wscript.Echo "Position:" & pos

'Positionsbestimmung abgeschlossen
'Ergebnis bzw. Position wird in Batchdatei (erg.bat)
'gespeichert
StreamOut.Write "SET POS="
StreamOut.Write pos
StreamIn.Close
StreamOut.Close
Wscript.Quit()
```

Anhang C – Quellcode runas.vbs

```
'RunAs Visual Basic Skript
'Author: Martin Hald ABB Schweiz AG
'Datum: 16.01.2004
'Syntax: runas.vbs <username> <password> <command>
'startet das als <command> Argument angegegebene Programm
'als Benutzer <username> mit dem <passwort>

set args=wscript.arguments
username = args(0)
pass = args(1)

'einlesen der uebergebenen Komanndozeilenargumente
for i=2 to args.count - 1
  command = command & args.item(i) & " "
Next

'Testausgabe der uebergeben Argumente
'wscript.echo "runas.exe /user:" & username & " """ & command
& """"

'Shell Objekt erzeugen
set shell=wscript.createobject("Wscript.shell")

'starten der runas.exe
shell.run "runas.exe /user:" & username & " """ &
               command & """"

'warten bis runas.exe komplett gestartet
wscript.sleep(1000)

'das Passwort und Enter an die Kommandozeile senden
shell.sendkeys pass & "{ENTER}"

'Skript Ende
wscript.Quit()
```

Anhang D – Quellcode startrun.exe

```
'Program startrun.exe
'Author: Martin Hald ABB Schweiz AG
'Datum: 26.01.2004
'Syntax: runas.vbs <username> <password> <command>
'startet das als <command> Argument angegegebene Programm
'als Benutzer <username> mit dem <passwort>

Funktionierende Quellcode:

     Private Sub Form_Load ()
          Dim oShell As Object
          Set oShell = GetObject ("", "Shell.Application")
          If ObjPtr(oShell) Then
               oShell.FileRun
          else
               oShell = CreateObject("Shell.Application")
               oShell.FileRun
          End if
          Unload Form1
     End Sub

---

Nicht funktionierende Quellcodes:

     `1.Möglichkeit
     Private Sub Form_Load ()
          Dim MyShell As New Shell
          MyShell.FileRun
          Unload Form1
     End Sub


     `2.Möglichkeit
     Option Explicit

     Private Declare Function SHRunDialog Lib "shell32" Alias _
          "#61" (ByVal hWnd As Long, ByVal hIcon As Long, _
          ByVal sPath As String, ByVal sTitle As String, _
          ByVal sPrompt As String, ByVal uFlags As Long)As Long

     Dim hWnd as Long

     Private Sub Form_Load ()
          SHRunDialog hWnd,0,0,vbNullString,vbNullstring,0
          Unload Form1
     End Sub
```

Anhang E – Quellcode runonce.bat

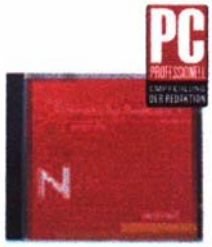
```
@echo off

        if %1 == get goto get
        if %1 == run goto run

        echo Wrong command-line switch. Try "get" or "run".
        dir x: 2> nul
        goto end


:get

        if exist %tmp%\ronce.in del %tmp%\ronce.in
        if exist %tmp%\ronce.out del %tmp%\ronce.out
        if exist %tmp%\ronceex.in del %tmp%\ronceex.in
        if exist %tmp%\ronceex.out del %tmp%\ronceex.out

rem *** Create the input files ***
        echo REGEDIT4> %tmp%\ronce.in
        echo.>> %tmp%\ronce.in
        echo
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnc
e]>> %tmp%\ronce.in
        echo.>> %tmp%\ronce.in

        echo REGEDIT4> %tmp%\ronceex.in
        echo.>> %tmp%\ronceex.in
        echo
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnc
eEx]>> %tmp%\ronceex.in
        echo.>> %tmp%\ronceex.in

rem *** Export the values from the registry ***
        regdat3 -e %tmp%\ronce.in %tmp%\ronce.out
        regdat3 -e %tmp%\ronceex.in %tmp%\ronceex.out

rem *** Remove the values from the registry ***
        regdat3 -df %tmp%\ronce.in
        regdat3 -df %tmp%\ronceex.in

rem *** Import empty keys into the registry ***
        regdat3 -i %tmp%\ronce.in
        regdat3 -i %tmp%\ronceex.in

        goto end


:run

        if not exist %tmp%\ronce.in goto error1
        if not exist %tmp%\ronce.out goto error1
```

```
        if not exist %tmp%\ronceex.in goto error1
        if not exist %tmp%\ronceex.out goto error1

        regdat3 -i %tmp%\ronce.out
        regdat3 -i %tmp%\ronceex.out


rem *** Check whether the system is RunOnceEx-able (since Internet
Explorer 4.0) ***
        if exist %windir%\system32\iernonce.dll goto DoRunOnceEx
        if exist %winbootdir%\system32\iernonce.dll goto DoRunOnceEx
        if exist %winpath%\system32\iernonce.dll goto DoRunOnceEx
        if exist %SystemRoot%\system32\iernonce.dll goto DoRunOnceEx
        if exist %windir%\system\iernonce.dll goto DoRunOnceEx
        if exist %winbootdir%\system\iernonce.dll goto DoRunOnceEx
        if exist %winpath%\system\iernonce.dll goto DoRunOnceEx
        if exist %SystemRoot%\system\iernonce.dll goto DoRunOnceEx

        goto DoRunOnce


:DoRunOnceEx
        start /w rundll32 iernonce.dll,RunOnceExProcess

:DoRunOnce
        if not "%os%" == "Windows_NT" start /w
%windir%\system\runonce -r
        if "%os%" == "Windows_NT" start /w runonce -r
     sleep32 1

        if exist %tmp%\ronce.in del %tmp%\ronce.in
        if exist %tmp%\ronce.out del %tmp%\ronce.out
        if exist %tmp%\ronceex.in del %tmp%\ronceex.in
        if exist %tmp%\ronceex.out del %tmp%\ronceex.out

        goto end


:error1
        echo The input files can't be found in the temporary folder.
        echo Did you execute "ronce.bat" with the switch "get"
before the reboot?
        dir x: 2> nul
        goto end


:end
```

Anhang F – Testbericht PC-Professional 06/2003

Produkt		Deployment Solution 5.5	Empirum Pro 9.0	Enterprise Management Suite 5.1
Hersteller		Altiris	Matrix 42	Baramundi
Gesamturteil	[Punkte]	befriedigend 75,0	gut 81,7	befriedigend 76,4
Deployment (30 %)	[Punkte]	69,0	78,6	72,9
Ausstattung (30 %)	[Punkte]	72,9	77,6	73,8
Bedienung (20%)	[Punkte]	72,0	84,0	72,0
Service (20 %)	[Punkte]	90,0	90,0	90,0
Info		(063 41) 91 75 90	(061 02) 81 60	(081 21) 56 70 80 20
Internet		www.altiris.com	www.matrix42.de	www.baramundi.de
Preis		54 Euro je User-Lizenz bis 99 Clients	ab 57 Euro je Client-Lizenz	10 Euro je OS-Deploymnet-Lizenz 15 Euro je Software-Deployment-Lizenz
Installation-Clients				
Komplette Erstinstallation		●	●	●
WOL/PXE/Partitionierung/Formatierung		●/●/○/○	●/●/●/●	●/●/●/●
Partitionen selektieren/Multiboot		●/●	●/○	○/○
OS-Installation: Windows 98/NT/2000/XP/ Unix/Linux		●/●/●/●/●/●	●/●/●/●/○/○	●/●/●/●/○/○
Nachträgliche OS-Konfiguration/Software		●/●	●/●	●/●
Deinstallation verteilter Pakete/ Deinstallation manueller Installation		●/○	●/○	●/○
Installation und Konfiguration Server				
Windows NT/2000/Unix/Linux		●/●/○/●	●/●/○/○	●/●/○/○
Datenhaltung MS SQL/Oracle/andere		●/○/MSDE	●/●/MSDE	●/○/○
Depotfähigkeit/Replikation automatisiert/ Replikation zeitgesteuert		●/●/●	●/●/●	●/●/●
Intelligente Bandbreitenunterstützung LAN/WAN/RAS/VPN		●/●/●/●	○/○/○/○	○/○/○/○
Lastverteilung auf mehrere Server/Loadbalancing		●/○	●/○	●/○
Schnittstelle zu ADS/NDS		●/○	●/●	●/○
Integration Verzeichnisdienste ADS/NDS		○/○	○/○	○/○
Schnittstellen Helpdesk/Remote/Inventarisierung		●/●/●	●/●/●	●/●/●
Monitoring/Meldungen und Reports		●/●	●/●	●/●
Verteilung per Unattended/Snapshot/MSI/ Native/Image		●/●/●/○/●	●/●/●/○/●	●/●/●/●/○
Administrator Rollenverteilung/zentral/dezentral		●/●/●	●/●/●	●/●/●
Sanfte Migration		●	●	●
Software-Verteilung				
Verteilung an Clients/User		●/○	●/●	●/○
Gruppierungen/Vererbung an Untergruppen		●/●	●/●	●/●
Parameter/Variablen in Paketen		○/●	○/●	○/●
Reboots definierbar		●	○	●
Wiederaufnahme bei Abbruch (intelligent)		○	●	○
Zeitsteuerung/Zeitfenster pro Pakete		●/●	●/●	●/●
Abhängigkeiten/Reihenfolge für Pakete		●/●	●/○	●/●
Ablehnen der Installation/Reminder/Installation im Hintergrund/gesicherte Installationsumgebung		●/●/●/●	●/●/●/●	●/●/●/●
Push-/Pull-Verfahren		●/●	●/●	●/●
Installation abhängig von Client-Eigenschaften		●	●	●
Service und Support				
Handbuch		englischsprachig, PDF	deutschsprachig, PDF	deutschsprachig, PDF
Support-Hotline		(063 41) 91 75 90	(061 02) 81 60	(081 21) 56 70 80 20
E-Mail		support@altiris-europe.com	support@matrix42.de	support@baramundi.de

● = ja ○ = nein



Landesk 6.62 Landesk befriedigend 76,3	Netinstall 5.56 Netsupport gut 83,4	On Command CCM 5.4 On Technology befriedigend 78,3	SMS 2.0 Microsoft befriedigend 65,3	Zenworks 4.0 Novell sehr gut 91,4
72,9	90,0	74,8	76,7	93,8
75,7	71,9	74,8	60,5	88,1
70,8	84,0	72,0	40,8	96,0
88,0	90,0	95,0	80,0	88,0
(089) 13 99 39	(07 11) 340 19 00	(081 51) 36 92 00	(018 05) 67 22 55	(02 11) 563 10
www.landesk.de	www.netsupport.de	www.ontechnology.com	www.microsoft.de	www.novell.de
75 Euro je Client-Lizenz	335 Euro für 10 Client-Lizenzen	Preise nur Projektbezogen	50 Euro je Client-Lizenz	74 Euro je Client-Lizenz
●	●	●	○	●
●/●/●/●	●/●/●/●	●/●/●/●	○/○/○/○	●/●/●/●
●/○	○/○	○/○	○/○	●/○
●/●/●/●/○/○	●/●/●/●/○/○	●/●/●/●/○/○	○/○/○/○/○/○	●/●/●/●/○/○
●/●	●/●	●/●	●/●	●/●
●/○	●/○	●/●	●/○	●/○
●/●/○/○	●/●/○/○	●/●/●/○	●/●/○/○	○/●/○/○ oder auf Netware ab 5.1
●/○/MSDE	●/●/MSDE	○/○/Sybase	●/○/○	●/●/My SQL
●/○/○	●/●/●	●/●/●	●/●/●	●/●/●
●/●/●/○	●/●/●/●	○/○/○/○	○/○/○/○	●/●/●/○
●/○	●/○	●/○	●/○	●/●
●/●	●/●	●/○	●/○	●/●
○/○	●/●	○/○	○/○	○/●
●/●/●	○/●/●	●/●/●	●/●/●	●/●/●
●/○	●/●	●/●	●/○	●/●
●/●/●/○/●	●/●/●/○/○	●/●/●/●/●	●/●/●/○/○	●/●/●/○/●
●/●/●	●/●/●	●/●/●	●/●/●	●/●/●
●	●	●	●	●
●/●	●/●	●/○	●/●	●/●
●/●	●/●	●/○	●/●	●/●
○/●	○/●	●/●	○/●	○/●
●	●	●	●	●
●	●	○	○	●
●/●	●/●	●/○	●/●	●/●
●/○	●/●	●/●, nur über Abhängigkeiten	●/○	●/●
●/●/●/●	●/●/●/●	●/●/○/●	●/●/●/●	○/○/●/●
●/●	○/●	●/●	●/○	●/●
●	●	●	●	●
deutschsprachig, HTML-Dokument auf CD-ROM	deutsch- und englischsprachig, PDF	englischsprachig, PDF deutsch-sprachig	deutschsprachig	deutschsprachig, PDF
(08 00) 624 35 26	(07 11) 34 01 90 48	(081 51) 36 92 00	(018 05) 67 22 55	(02 11) 563 10
infoeurope@landesk.com	info@netsupport.de	über Homepage: www.ontechnology.de/daten/services/index.html	über Homepage: Support.microsoft.com/de	informationen@novell.com

Anhang G – Treiberintegration

Erweiterung config.swp:

```
#------------------MH Copy Drivers-------------------#

echo copy pnp drivers.....
#create main directories
if not exist %WINDRV%\drivers md %WINDRV%\drivers    > NUL
if not exist %WINDRV%\drivers\nic   md %WINDRV%\drivers\nic    > NUL
if not exist %WINDRV%\drivers\video md %WINDRV%\drivers\video    > NUL
if not exist %WINDRV%\drivers\sound md %WINDRV%\drivers\audio    > NUL
if not exist %WINDRV%\drivers\modem md %WINDRV%\drivers\modem    > NUL
if not exist %WINDRV%\drivers\other md %WINDRV%\drivers\other    > NUL
if not exist %WINDRV%\drivers\chipset md %WINDRV%\drivers\chipset    > NUL

#check if a pctype is selected and set the corresponding variables by the
#batch file
if "%sel_pctype%" == "none" goto end_pctype
     if "%sel_pctype%" == "select" goto driver_selected
            smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\pctype\%sel_pctype%.bat
                          %WINDRV%\drivers\
            call %WINDRV%\drivers\%sel_pctype%.bat

:driver_selected

#check which driver to copy and copy it into client directory video
if "%video1%" == "none" goto end_v1
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\video\%VIDEO1%\*.*
                    %WINDRV%\drivers\video\video1
:end_v1
if "%video2%" == "none" goto end_v2
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\video\%VIDEO2%\*.*
                    %WINDRV%\drivers\video\video2
:end_v2
#selbe Art der Einträge für die restlichen Grafikkarten

#nic
if "%nic1%" == "none" goto end_n1
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\nic\%NIC1%\*.*
                    %WINDRV%\drivers\nic\nic1
:end_n1
if "%nic2%" == "none" goto end_n2
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\nic\%NIC2%\*.*
                    %WINDRV%\drivers\nic\nic2
#selbe Art der Einträge für die restlichen Netzwerkkarten

#modem
if "%modem1%" == "none" goto end_m1
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\modem\%MODEM1%\*.*
                    %WINDRV%\drivers\modem\modem1
:end_m1

#audio
if "%sound1%" == "none" goto end_s1
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\audio\%SOUND1%\*.*
                    %WINDRV%\drivers\audio\audio1
:end_s1
```

```
#chipsets
if "%chipset1%" == "none" goto end_c1
       smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\chipset\%CHIPSET1%\*.*
                  %WINDRV%\drivers\chipset\chipset1
:end_c1
if "%chipset2%" == "none" goto end_c2
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\chipset\%CHIPSET2%\*.*
                  %WINDRV%\drivers\chipset\chipset2
:end_c2

#other
if "%other1%" != "none" goto end_o1
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\other\%OTHER1%\*.*
                  %WINDRV%\drivers\other\other1
:end_o1
if "%other2%" != "none" goto end_o2
      smecp16 -s -f \\depot\%W2KPDEPOT%\5\pnp\other\%OTHER2%\*.*
                  %WINDRV%\drivers\other\other2
:end_o2

:end_pctype
echo ....pnp drivers copied

#----------------END MH PNP ----------------------------#
```

Definition eines PC-Typs:

```
REM ---------- PNP-Drivers for Transtec 1300 PC ------------
set NIC1=i82550
set NIC2=none
...
set VIDEO1=matxg450
set VIDEO2=nvidgefo
set VIDEO3=none
...

set SOUND1=none

set MODEM1=none

set OTHER1=none
set OTHER2=none

set CHIPSET1=intel
```

Literaturverzeichnis

[1] Laura Lemay & Rogers Cadenhead, *JAVA 2*, Markt & Technik Verlag 1999

[2] Ralph Steyer, *Java 2*, Markt & Technik Verlag 1999

[3] Yves Lepage & Paul Iarrera, *Die UNIX Sysad Bibel*, mitp Verlag 1999

[4] Microsoft Press, *Grundlagen des Netzwerkbetriebs 2.Auflage*, 1997

[5] Suse Linux 9.0, *Administrationshandbuch*, 2003

[6] Suse Linux 9.0, *Benutzerhandbuch*, 2003

[7] Dr. Bernhardt Röhrig, *Linux im Netz*, C&L Verlag 1997

[8] Thomas Dubbert & Dominik Bergeau, *LINUX NEW REFERENCE,* Markt & Technik Verlag 2000

[9] Suse Homepage Autoyast, http://www.suse.de/~nashif/autoinstall

[10] Microsoft Homepage, http:// www.microsoft.com

[11] CHIP Forum, www.chip.de, *FAQ - Raid*

[12] PC-Professional Ausgabe 06/2003 Netzwerkspecial „Deployment-Programme“

[13] Handbuch OnTechnology, On Command Erste Schritte, 2003

[14] Handbuch OnTechnology, On Command CCM Referenz, 2003

[15] Handbuch OnTechnology, On Command Referenz für Paketentwickler, 2003

[16] Handbuch OnTechnology, On Command CCM Package Import Tool, 2003

[17] Homepage OnTechnology, www.ontechnology.de

[18] Homepage W3C „Extensible Markup Language (XML)“, www.w3.org/xml

[19] Microsoft Homepage “Microsoft Windows 2000 Guide to Unattended Setup “, www.microsoft.com/technet/prodtechnol/windows2000pro/deploy/unattend/sp1unatd.asp

[20] HP Homepage “ignite-ux – summary”,http://www.software.hp.com/products/IUX/

[21] Microsoft Homepage “155614 - Unattended Installation of Microsoft Windows NT 4.0”, http://support.microsoft.com/support/kb/articles/Q155/6/14.asp

Abbildungs- und Tabellenverzeichnis

Abbildung 1 - Phasen während der Betriebsysteminstallation 10
Abbildung 2 - Ablauf einer RIS Installation 15
Abbildung 3 - Kontrollzentrum „Automatische Installation“ 23
Abbildung 4 - autoinst.xml mit YaST erstellen 24
Abbildung 5 - PC Professional Testergebnis Details 31
Abbildung 6 - PC Professional Gesamt 32
Abbildung 7 - Remote Method Invocation 35
Abbildung 8 - Wo startet CCM ? 37
Abbildung 9 - Raid Level 0 41
Abbildung 10 - Raid Level 1 42
Abbildung 11 - Raid Level 2 43
Abbildung 12 - Raid Level 3 44
Abbildung 13 - Raid Level 4 45
Abbildung 14 - Raid Level 5 46
Abbildung 15 - Raid Level 10 47
Abbildung 16 - Raid-Konfiguration (RaidWatch) 48
Abbildung 17 - Typische CCM-Umgebung 55
Abbildung 18 – Netzwerkstruktur 58
Abbildung 19 - MSI Wizard Startbildschirm 62
Abbildung 20 - MSI Wizard Grundeinstellungen 63
Abbildung 21 - MSI Wizard Standardproperties 64
Abbildung 22 - MSI Wizard Advanced Properties 65
Abbildung 23 - MSI Wizard Installationsoptionen 66
Abbildung 24 - MSI Wizard Paketerzeugung beendet 67
Abbildung 25 - MSI Wizard Paket kopieren 68
Abbildung 26 - MSI Wizard Paket speichern 69
Abbildung 27 - MSI Wizard Paket fertig 69
Abbildung 28 - Development Studio Erste Schritt 73
Abbildung 29 - Development Studio Projektein 74
Abbildung 30 - Development Studio Aufzeichnung starten 74
Abbildung 31 - Development Studio Aufzeichnung 75
Abbildung 32 - Development Studio Aufzeichnung beenden 76

Abbildung 33 - Development Studio Transcript 77
Abbildung 34 - Development Studio WIN-File 78
Abbildung 35 - Development Studio Skripterstellung abgeschlossen 79
Abbildung 36 - Development Studio Skript kompilieren 80
Abbildung 37 - Development Studio Skript testen 81
Abbildung 38 - Development Studio Config.swp erstellen 82
Abbildung 39 - Package Import Tool 84
Abbildung 40 - Positionsproblem (Extensions) 93
Abbildung 41 - Windows Start -> Ausführen 96
Abbildung 42 - CCM Hardware- und Softwareprofil 103
Abbildung 43 – CCM Schnittstellen 104

Tabelle 1 - Vor/Nachteile Installationsmethoden 9
Tabelle 2 - Vergleich Sysprep,RIS & unattended Setup 18
Tabelle 3 - RPM Installationsoptionen 30
Tabelle 4 - RAID Konfiguration 49
Tabelle 5 - In-Band SCSI Raid Einstellungen 50
Tabelle 6 - Microsoft Windows 2000 Server Security Patches [10] 53
Tabelle 7 - Microsoft Windows 2000 Server Funktionspatches [10] 53
Tabelle 8 - Wichtige CCM Dateien 60

Abkürzungsverzeichnis

ACPI	Advanced Configuration and Power Interface	Schnittstelle zur Integration von Energieverwaltungsfunktionen
API	Application Programming Interface	Schnittstellen(-beschreibung) zur Anwendungsprogrammierung
ADK	Application Developers Kit	OnTechnology CCM API Paket
DHCP	Dynamic Host Configuration Protocol	Protokoll zur dynamischen Vergabe der Netzwerkkonfiguration
DLL	Dynamic Link Library	Bibliothek zur Laufzeit eines Programms gelinkt wird
JRE	Java Runtime Environment	Java Virtuelle Maschine von SUN die zur Ausführung von Java Programmen benötigt wird
MSI	Microsoft Installer	Datenbankdatei die dir für das Setup einer Anwendung benötigten Informationen enthält
RIS	Remote Installation	Windows 2000 Installationsmethode
RPM	Redhat Package Manager	Installationsmethode unter Suse Linux Vergleichbar mit MSI
RMI	Remote Method Invocation	Java Standard für verteilte Objekte
SID	Security Identifier	Sicherheitsbezeichner der einen Benutzer oder eine Gruppe kennzeichnet
SMS	Short Message Service	Kurznachrichtendienst via GSM
UDF	Uniqueness Database Files	wird beim unattended Setup für unterschiedliche Konfigurationen unter Windows NT verwendet
WoL	Wake up on Lan	Rechnerfunktion, die das Aufwecken
XML	Extensible Markup Language	Erweiterbare Textauszeichnungsprache
YaST	Yet another System Tool	Setupprogramm von Suse Linux